www.farelettronica.com

N° 267 SETTEMBRE - Anno 23 - € 6,00

speciale

PROGETTARE UN SISTEMA EMBEDDED USANDO I COMPUTER-ON-MODULE

LINGUAGGI:

LADDER LOGIC

Attraverso alcuni esempi pratici approfondiamo il registro F e la gestione dei dati

NCQ

Introduzione al linguaggio NQC utilizzato

nella programmazione dei robot Lego Mindstorms

SPIN

Introduzione al linguaggio SPIN utilizzato nel microcontrollore Parallax Propeller

DISPLAY LCD GRAFICI

Realizziamo le prime librerie in linguaggio C per il controllo dei display grafici

La progettazione di un adattatore di impedenza con BJT, FET e Mosfet

Tutto sui TRASDUTTORI DI POSIZIONE come funzionano potenziometri, trasformatori differenziali, syncro

Realizzazioni pratiche con

TEXAS MSP 430

Generatori di onde quadre, un convertitore AD sigma/delta, monitor RS232, uso del watchdog timer

TELECAMERA MOTORIZZATA

PROGETTO completo PER UN CONTROLLO PAN&TILT

IDEE DI PROGETTO

- pilotare la porta parallela del PIC
- generatore di CORRENTE COSTANTE
- amplificatore audio da 68W
- divisore per n con 4017
- messaggio SOS

ISSN 1591-2272

MENSILE Poste Italiane Spa - Spedizione in abbonamento Postale - D.L. 353/2003 (conv. In L. 27/02/2004 n. 46) art. 1, comma1, DCB Milano. In caso di mancato recapito, restituire all'editore che si impegna a pagare la relativa tassa presso il CPM di Roserio - Milano

Richiedi maggiori informazioni sui contenuti di Fare Elettronica, visita il sito:

www.farelettronica.com/mip

Oppure

compila questo modulo ed invialo via fax al numero 02 66508225

Numero della rivista 267

Codici MIP* (da compilare)

DATI DEL LETTORE

Nome

Cognome

Azienda

Indirizzo Azienda

Cap | Città

Prov. | Nazione

Tel.

Fax

e-mail

Privacy. Ai sensi del Decr. Lgs. 196/2003 la informiamo che i dati trasmessi verranno impiegati coi principali scopi di indagini di mercato e nelle modalità previste dallo stesso, prevalentemente con mezzi informatici. Il conferimento, di norma facoltativo, è obbligatorio per permettere il rapporto commerciale. È in ogni caso fatto diritto dell'interessato esercitare i propri diritti, nei modi previsti dal "Titolo II art. 7" della legge sopra citata, scrivendo a Inware Edizioni Via Cadorna 27 – 20032 Cormano o trmite email a info@inwaredizioni.it

* Utilizza il numero MIP che compare alla fine di ogni articolo o all'interno delle pagine di tuo interesse

Guida al numero 267

Informati!

Richiedi maggiori informazioni sui contenuti di Fare Elettronica, visita il sito: www.farelettronica.com/mip

pag. 3

Rispondi&Vinci!

Divertiti e metti alla prova le tue conoscenze con Elettro Quiz e vinci ogni mese fantastici premi!

pag. 70

Risparmia!

Abbonati o Rinnova oggi il tuo abbonamento a Fare Elettronica

1 ANNO × 11 RIVISTE = €49,50

CON UN RISPARMIO DEL 25%

pag. 112

Pratica

SETTEMBRE 2007

Teoria

Speciale

Aziende citate in questo numero

Risorse

www.farelettronica.com

DIRETTORE RESPONSABILE
Antonio Cirella

DIRETTORE ESECUTIVO
Tiziano Galizia

COORDINAMENTO TECNICO
Maurizio Del Corso

HANNO COLLABORATO IN QUESTO NUMERO
Giovanni Di Maria, Savino Giusto, Antonio Di Stefano, Nico Grilloni, Giuseppe Modugno, Grazia Ancona, Roberto Verzino, Giuseppe La Rosa, Fabio Riscica.

DIREZIONE • REDAZIONE • PUBBLICITÁ
INWARE Edizioni srl - Via Cadorna, 27/31 - 20032 Cormano (MI)
Tel. 02.66504755 - Fax 02.66508225
info@inwaredizioni.it - www.inwaredizioni.it
Redazione: fe@inwaredizioni.it

INTERNATIONAL ADVERTISEMENT
IEM - Wintergasse, 52- 3002 Purkersdorf Austria
Tel. +43 2231 68347 - Fax. +43 2231 68402
IEM@inwaredizioni.it

GRAFICA E IMPAGINAZIONE
Margherita Nardelli

STAMPA
ROTO 2000 - Via L. da Vinci, 18/20 - 20080, Casarile (MI)

DISTRIBUZIONE
Parrini & C. S.p.a. - Viale Forlanini, 23 - 20134, Milano

UFFICIO ABBONAMENTI
INWARE Edizioni srl - Via Cadorna, 27/31 - 20032 Cormano (MI)
Per informazioni, sottoscrizione o rinnovo dell'abbonamento:
abbonamenti@inwaredizioni.it
Tel. 02.66504755 - Fax. 02.66508225
L'ufficio abbonamenti è disponibile telefonicamente
dal lunedì al venerdì dalle 14,30 alle 17,30
Tel. 02.66504755 - Fax. 02.66508225
Poste Italiane S.p.a. - Spedizione in abbonamento Postale
D.L. 353/2003 (conv. In L. 27/02/2004 n. 46) art. 1, comma1, DCB Milano.
Abbonamento per l'Italia: € 49,50 - Abbonamento per l'estero: € 115,00
Gli arretrati potranno essere richiesti, per iscritto, a € 9,00
oltre le spese di spedizione

Autorizzazione alla pubblicazione del Tribunale di Milano n. 647 del 17/11/2003.

RICHIESTE DI ASSISTENZA
Per richiedere assistenza o chiarimenti sugli articoli pubblicati, vi preghiamo di utilizzzare il servizio MIP compilando l'apposito modulo on-line all'indirizzo www.farelettronica.com/mip.

COLLABORARE CON FARE ELETTRONICA
Le richieste di collaborazione vanno indirizzate all'attenzione di Tiziano Galizia *(t.galizia@inwaredizioni.it)* e accompagnate, se possibile, da una breve descrizione delle vostre competenze tecniche e/o editoriali, oltre che da un elenco degli argomenti e/o progetti che desiderate proporre.

ELENCO INSERZIONISTI

Note dalla redazione

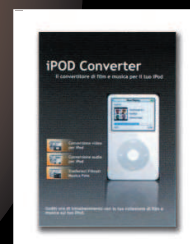
Tiziano Galizia
t.galizia@inwaredizioni.it

Tatto o voce nell'interfaccia utente del futuro?

Chi come me ha figli, conosce quella splendida sensazione che si prova quando i bimbi cercano di interagire per la prima volta. Ti guardano con quello sguardo tra l'incredulo e il sorpreso e cercano di toccarti la faccia, elevando così il tatto a senso primario, almeno per quella età. È innegabile che il tatto sia il metodo più semplice di interagire con un dispositivo, lo dimostra il fatto che tutte le aziende stanno investendo in questa direzione. Basti pensare ad Apple con il suo iPhone e Microsoft con la tecnologia Surface.
"Multitouch", un modo nuovo di controllare i prodotti, utilizzando uno schermo sensibile al tocco di più dita, o più utenti, contemporaneamente.
Bello, anzi fantastico, ma inutilizzabile mentre si cerca di fare altro. Io in auto utilizzo un kit viva-voce, ma per comporre il numero devo ancora utilizzare la tastiera del telefonino, anche se ormai riconosco i tasti a occhi chiusi. Devo comunque cercare il telefono e un tasto da prendere come riferimento. Ora immaginate di fare la stessa cosa ma con un telefono dotato di LCD sensibile al tocco, è impossibile, dovete fermare il veicolo e guardare il telefono per capire dove andare a toccare.
La soluzione c'è e si chiama Riconoscimento Vocale, tecnologia della quale sono un convinto sostenitore.
Lo stato attuale di questa tecnologia è molto buono, nel senso che si è raggiunto un ottimo livello di riconoscimento, anche in ambienti rumorosi. Nel mercato consumer, infatti, sempre più prodotti utilizzano questa tecnologia.
Tempo fa ho comprato una sveglia da viaggio interamente controllata a voce, mi basta chiedergli che ora è in California e lei mi risponde immediatamente. Bello vero? Semplice e immediato.
Come ho scritto in un mio precedente editoriale, è il riconoscimento vocale l'interfaccia utente del futuro.
È una tecnologia che sta facendo passi da gigante, un mercato in continua evoluzione dove le grandi aziende stanno facendo acquisti a suon di milioni, lo dimostra la recente acquisizione di Voice Signal da parte di Nuance per un valore di gran lunga più alto del reale valore dell'azienda. Il prezzo pagato ha dell'incredibile considerando che stiamo parlando di una tecnologia ancora considerata (a torto) "emergente".
Il consiglio che posso dare ai lettori più giovani è sicuramente quello di approfondire questa tecnologia. Sono convinto che nei prossimi anni ci sarà una grandissima richiesta di tecnici in questo campo.

Vi auguro una piacevole lettura e vi rinnovo l'appuntamento in edicola a Ottobre.

Rubrica di notizie e novità dal mondo dell'elettronica.

Prima pagina

ERNI Electronics acquisiSCe Pewamat Elektronik GmbH

ERNI Electronics GmbH ha acquisito Pewamat Elektronik GmbH di Zell/Aichelberg, Germania. Le attività correnti della società sono proseguite in modo indipendente sotto il nuovo marchio ERNI ES GmbH. L'azienda è guidata da Gerald Römpp, in precedenza responsabile vendite "Systems" per ERNI Electronics GmbH. Grazie all'acquisizione di Pewamat, ERNI Electronics ha ottenuto l'accesso ad una moderna tecnologia manifatturiera e di testing orientata alla produzione di schede attive, proseguendo così l'opera di ampliamento della sua offerta nel campo dei servizi "Electronic Solutions".
Grazie all'ingresso di questo know-how nel settore delle tecnologie di saldatura, ERNI può offrire un ampio spettro di schede attive, subassiemi e sub-rack, proponendosi come fornitore unico. Oltre al completo portafoglio di connettori, la società mette a disposizione anche i corrispondenti apparati di assemblaggio e di macchinari di saldatura per SMD e ad onda, nonché i relativi ausili di test. ERNI ES GmbH dispone di due moderne linee di saldatura per il trattamento di componenti through-hole (THT) in processi esenti da piombo (o, in alternativa, con piombo).
Parallelamente, ERNI ha anche raddoppiato le capacità di saldatura SMT grazie ad una serie di macchine di assemblaggio ad alte prestazioni, e la società può produrre in modo efficiente PCB in grandi o piccoli lotti. L'uso di particolari feeder, permette di trattare senza problemi componenti di grosse dimensioni o ad elevato pin-count (quali i connettori SMC a 80-pin), e l'uso di tester in-circuit ad alte prestazioni e di altri apparati specializzati di verifica garantisce elevati livelli di funzionalità e di qualità.

Codice MIP 267201

In mostra al Nab l'offerta di interfacce video professionali di Cypress Semiconductor

Nel corso della recente edizione della National Association of Broadcasters 2007, Cypress Semiconductor ha presentato il proprio portafoglio di soluzioni di interfaccia per apparecchiature video professionali. Tra i prodotti di punta si possono annoverare i seguenti:

- Driver per cavi operanti a 3 Gbps per la trasmissione di segnali video;
- Una scheda dimostrativa che utilizza il SERDES HOTLink® II e un equalizzatore video che permette di realizzare una soluzione di trasmissione;
- Il design di riferimento Video-over-IP di Altera Corporation che utilizza i driver per cavi e l'equalizzatore video di Cypress Semiconductor.

La famiglia di SERDES HOTLink di Cypress Semiconductor mette a disposizione interfacce di comunicazione seriali di tipo punto-punto o punto-multipunto per collegamenti ottici o in rame ad alta velocità. Tra le caratteristiche salienti di questi componenti si possono annoverare serializzazione e deserializzazione, codifica/decodifica 8B/10B opzionale e funzioni di framing. Grazie alla possibilità di variare la frequenza tra 50 Mbps e 1,5 Gbps, i SERDES HOTLink possono essere impiegati per la realizzazione sia di protocolli di comunicazione seriale proprietari sia dei protocolli tradizionali quali Fiber Channel, DVB-ASI, Gigabit Ethernet (GigE), ESCON®, SMPTE 259M e SMPTE 292M.
Gli equalizzatori video a più velocità (HD/SD/DVB-ASI) di Cypress Semiconductor sfruttano la tecnologia di equalizzazione proprietaria CLEANLink™ per assicurare elevate prestazioni nei sistemi video professionale. Per soddisfare

le specifiche DVB-ASI in applicazioni quali amplificatori per la distribuzione video, commutatori e router per la produzione, convertitori A/D e D/A e altre apparecchiature di broadcasting professionali, gli utenti possono utilizzare i driver per cavi multi-rate di Cypress Semiconductor per i segnali DVB-ASI non invertenti senza dover aggiungere dispositivi esterni.

Codice MIP 267202

Atmel Annuncia un Transceiver WiMAX a Basso Costo

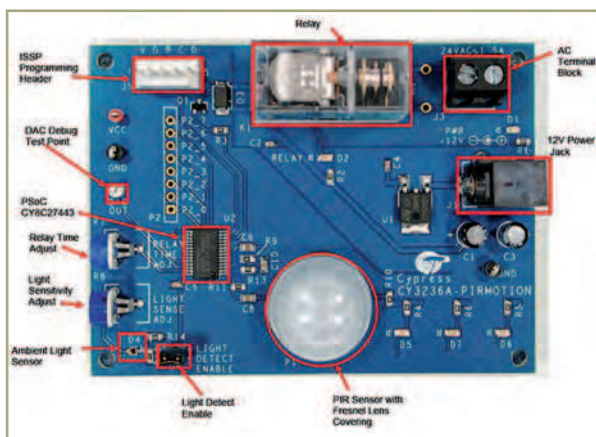
Atmel® Corporation annuncia il secondo transceiver della famiglia MAX-Link™ progettato per le applicazioni WiMAX. Oltre alla funzionalità WiMAX, l'AT86RF535B consente di raggiungere livello di costo particolarmente contenuti tra quelli presenti sul mercato. Questo transceiver a singolo chip frequency-agile opera a 3,5GHz con opzioni di indirizzamento a banda passante multipla. La configurazione interna e i registri di controllo possono essere programmati per integrarsi con i dispositivi di numerosi costruttori. Tutti i transceiver MAX-Link offrono eccellenti prestazioni RF con consumi particolarmente ridotti, basso costo, e un numero ridotto di componenti esterni. In dispositivo contiene un amplificatore a basso rumore, un driver per l'amplificatore di potenza, il mixer del transceiver, i filtri del trasmettitore e ricevitore, e il sistema di controllo della potenza del trasmettitore, funzioni tutte controllate digitalmente. Il transceiver può essere configurato per operare con valori di IF o direttamente in banda base senza l'aggiunta di filtri esterni. Tutto il sistema comprensivo di antenna diversity e circuiti di alimentazione utilizza un'area di soli 14cm^2. L'AT86RF535B utilizza un sistema di calibrazione che ottimizza la reiezione della frequenza immagine e il leakage dell'oscillatore locale. Il sintetizzatore utilizza una interpolazione di fase che consente una elevata risoluzione della frequenza e una sintonia veloce.

Codice MIP 267203

Austriamicrosystems e NemeriX annunciano la spedizione di 5 milioni di dispositivi GPS RF

Austriamicrosystems, una azienda leader per la progettazione e la produzione di circuiti integrati analogici specializzata in soluzioni a basso consumo e soluzioni GPS per la localizzazione, annuncia la spedizione del 5 milionesimo dispositivo NemeriX NJ1006A RF. Questo dispositivo ha contribuito a garantire un grande successo a NemeriX nei mercati dei dispositivi per la navigazione personale (PND) e di Bluetooth, dove le piccole dimensioni e l'elevata sensibilità costituiscono un fattore chiave per il successo. Il NemeriX NJ1006A è un front-end RF a basso rumore destinato all'impiego nei sistemi di Global Positioning Systems (GPS) destinati all'impiego nelle applicazioni portatili e automotive. Grazie all'integrazione di un amplificatore a basso rumore e al circuito risonante dell'oscillatore, si ottiene una sostanziale riduzione dei componenti e delle dimensioni fisiche del prodotto, unitamente ad eccellenti prestazioni. Il dispositivo supporta i clock standard dei sistemi UMTS, EDGE, GPRS/GSM, CDMA, rendendolo adatto anche per le applicazioni di telefonia mobile.

Codice MIP 267204

Kit di valutazione PSoC™ per il rilevamento del movimento di tipo pir e l'espansione della porta I²C

Cypress Semiconductor introduce due nuovi kit di valutazione mixed-signal arrays Programmable System-on-Chip™ (PSoC™). Questi kit, il primo per il rilevamento del moto di tipo PIR (Pyroelectric InfraRed) e il secondo per l'espansione della porta I²C, sono strumenti di uso estremamente semplice che permettono di completare in tempi rapidi e in maniera efficace progetti che contemplano l'uso dei dispositivi PSoC destinati a una molteplicità di applicazioni. Queste, in sintesi, le caratteristiche salienti dei nuovi kit:

- CY3236A-PIRMOTION – Contiene hardware e software necessari per implementare il rilevamento del movimento utilizzando un sensore di movimento di tipo PIR controllato in maniera "intelligente" da CY8C27443, un dispositivo della serie PSoC. Il kit include la scheda del sensore di movimento PIR, un alimentatore a 12 V, il CD con l'ambiente di sviluppo integrato (IDE) PSoC Designer™, e un CD con schemi circuitali, lista dei materiali (BOM), file Gerber e un design

esemplificativo realizzato con PSoC Designer. Il kit può essere impiegato per il controllo automatico dell'illuminazione, apriporta automatici, sistemi di sicurezza, apertura di chioschi e attivazione di telecamere wireless.

- CY3242-IOXLite: Contiene hardware e software necessari per realizzare l'espansione di I/O del bus I^2C, la gestione del sistema e il controllo del pannello frontale sfruttando un solo dispositivo PSoC per l'espansione della porta I^2C, il CY8C9520. Il kit contiene una scheda dimostrativa per un expander I^2C, un cavo USB, la necessaria documentazione e un CD con i design di progetto. Tra le applicazioni tipiche si possono annoverare server PC, console personali per videogiochi, apparecchiature di memorizzazione e networking.

Codice MIP 267205

Micron presenta due nuovi prodotti di ultima generazione

Micron annuncia due nuovi prodotti di ultima generazione per applicazioni nel settore dell'elettronica di consumo. Si tratta del più veloce chip di memoria da 1Gigabit e della memoria NAND Flash più innovativa del mercato. Il nuovo chip, una memoria DDR2 (Double Data Rate) da 1 gigabit, è in grado di gestire fino a 1066 megabit al secondo e viene utilizzato principalmente in applicazioni ad elevata intensità di dati, quali l'elaborazione di materiali video o i videogiochi di alta qualità. I moduli di memoria DDR2-1066 della Micron permettono una fruizione ottimale da parte dell'utente e rendono le immagini su schermo molto più realistiche. La DDR2-1066 offre un notevole miglioramento della velocità, con un aumento del 33% rispetto alla memoria DDR2-800, attualmente la più veloce in uso.
La nuova memoria NAND Flash nasce dalla collaborazione con Intel con la quale Micron ha costituito nel 2005 una joint venture per collaborare allo sviluppo e alla produzione delle memorie Flash. Il nuovo prodotto è una memoria multilivello, una memoria cioè sviluppata su due piani, che permette di raddoppiare il numero di bit a parità di superficie. La tecnologia utilizzata per la fabbricazione permette di immagazzinare e gestire moltissimi dati in pochissimo spazio: le strutture di questa nuova memoria hanno infatti dimensioni nell'ordine dei 50 nanometri.

Codice MIP 267206

INCOSE sceglie Telelogic DOORS per la gestione di documenti

Telelogic ha annunciato che l'International Council on Systems Engineering (INCOSE) ha scelto Telelogic DOORS® per la gestione dei requisiti. L'accordo include 20 licenze, che saranno utilizzate dal Comitato pianificazione e controllo, dal CdA ed altri team del Consorzio di INCOSE.
INCOSE utilizzerà Telelogic DOORS per gestire e tracciare i requisiti di documenti di pianificazione strategici, che guidano lo sviluppo organizzativo in ottemperanza alle "best practice" per quanto riguarda la modellazione dei sistemi. Il pacchetto include la gestione e l'organizzazione di progetti a lungo termine (15-20 anni), un piano strategico per 5 anni ed un sistema di pianificazione e controllo. Telelogic DOORS facilita ed accelera lo sviluppo e la gestione dell'informazione, fornendo ai soci di INCOSE un accesso veloce nell'allocazione di specifici requisiti. Telelogic installerà DOORS sul database di INCOSE attraverso un server remoto sito in California, a cui INCOSE ha accesso tramite una "Web interface". Prima di implementare DOORS, la preparazione dei documenti utilizzava fogli elettronici, ma il tempo dedicato all'aggiornamento dei dati era troppo dispendioso e i dettagli delle decisioni strategiche erano difficili o impossibili da schematizzare.

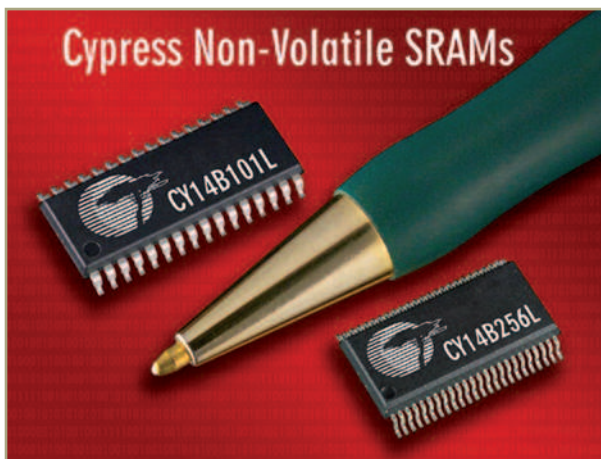
Codice MIP 267207

Fujitsu annuncia caratteristiche e disponibilità della tecnologia Mobile WiMAX System-on-Chip

Fujitsu Microelectronics ha annunciato caratteristiche e piani di consegna riguardanti il primo SoC (System-on-Chip) baseband mobile della società, che incarna la nuova frontiera dell'innovazione nel campo delle soluzioni WiMAX su silicio.

Il chip Fujitsu MB86K21 802.16e fornirà sostanziali guadagni prestazionali e di velocità rispetto alle alternative concorrenti, unitamente a consumi ridotti, con un fattore forma estremamente compatto.
Il nuovo SoC sarà il primo esponente della categoria a essere sottoposto al processo di certificazione WiMAX Forum Wave 2.
Le prime campionature del nuovo SoC saranno disponibili a partire da agosto 2007. Questo SoC baseband mobile altamente integrato su unico chip, per la cui produzione viene utilizzata la tecnologia di processo proprietaria da 90nm a bassa dispersione, si propone come soluzione ideale per PC card e dispositivi multimediali mobili. Nell'ambito di test di laboratorio e installazioni di prova, il nuovo SoC mixed-signal ha raggiunto una velocità di trasferimento pari a 18,6Mbps. Il System Design Kit (SDK) e il PC Card Reference Platform Delivery Kit (RDK) sono progettati per aiutare i clienti a giungere rapidamente alla produzione di massa.
Entrambi i kit sono corredati da pacchetti hardware e software di riferimento completi. L'SDK comprende interfacce USB, PC card e SDIO (Secure Digital Input/Output) per consentire ai clienti di collegare facilmente le loro piattaforme.

Codice MIP 267208

Le SRAM non volatili di Cypress Semiconductor vincono il premio ACE di EE Times China

Cypress Semiconductor ha annunciato che le proprie SRAM di tipo non volatile (nvSRAM) sono state dichiarate Prodotto dell'Anno nella categorie "Memorie" prevista dal concorso ACE (Annual Creativity in Electronics) organizzato dalla rivista EE Times-China. Il riconoscimento "Prodotto dell'Anno" viene assegnato a dispositivi o tecnologie che hanno avuto un impatto significativo sul mercato elettronico cinese o ne hanno tutte le premesse. I vincitori sono stati scelti da un gruppo di esperti appartenenti al mondo industriale.
Le nvSRAM di Cypress Semiconductor immagazzinano i dati in assenza di alimentazione senza richiedere la presenza di una batteria. L'eliminazione della batteria esterna consente a Cypress Semiconductor di proporre dispositivi ad elevata velocità conformi alle direttive RoHS in package di dimensioni più ridotte rispetto alle alternative disponibili sul mercato. Le nvSRAM sono i componenti ideali per tutte le applicazioni che richiedono una scrittura dei dati a elevata velocità e la assoluta sicurezza dei dati immagazzinati in maniera non volatile, come ad esempio RAID (Redundant Array of Independent Disk), fotocopiatrici, terminali POS, misuratori palmari e prodotti elettronici per il mercato consumer. Queste memorie nvSRAM sono la prima famiglia di dispositivi sviluppati in seguito all'accordo stipulato tra Cypress Semiconductor e Simtek Corporation, azienda che ha inventato e lanciato il concetto di nvSRAM.

Codice MIP 267209

Austriamicrosystems amplia il portafoglio di memorie non volatili

Austriamicrosystems annuncia una ulteriore espansione del suo portafoglio di memorie non volatili con l'introduzione di un nuovo blocco EEPROM caratterizzato da una affidabilità di livello superiore derivante dall'impiego del processo da 0,35µm. Il nuovo blocco da 3kBit oggi disponibile è completamente qualificato per le applicazioni automotive ed è adatto ad operare in ambienti ostili, facendone cosi la scelta ideale per le applicazioni automotive, medicali e industriali.
Molti prodotti come i sistemi di telelettura, le unità per controllo motori, i sistemi di gestione dei motori traggono beneficio da un elevato livello di affidabilità su una gamma estesa di temperatura che arriva a 180°C. Con una durata tipica di 20 milioni di cicli di lettura/scrittura a 25°C e 1 milione di cicli garantiti a 125°C, mentre la ritenzione dei dati è garantita per 25 anni a 125°C.
Il blocco di memoria utilizza una tensione di alimentazione compresa tra 2,2V e 3,6V per la lettura, e 2,5V e 3,6V per la scrittura è gia disponibile.

Codice MIP 267210

Soluzione completa per

Apertura a Distanza

Atmel® fornisce una soluzione completa e sicura basata su AES (Advanced Encryption Standard) Customizza i tuoi progetti con i nostri tools RAC e la nota di applicazione gratuita

Sistema RAC

RAC sta per Remote Access Control e consiste nel controllo remoto di diverse applicazioni.
E' spesso descritto come Remote Keyless Entry o Remote Wireless Entry.
Un sistema è tipicamente composto da un ricevitore e uno o più trasmettitori.
Essendo portatili, sono alimentati a batteria.
I sistemi RAC operano su bande radio libere (ISM) e le caratteristiche di sicurezza e basso consumo sono i due aspetti più critici.

Tecnologia picoPower™

- Micro AVR® standard ad alte prestazioni
- Maggiore durata delle batterie
- Numero ridotto di componenti esterni
- Massima flessibilità con un solo micro

Sicurezza

- Basata sul robusto algoritmo AES
- Trasmissioni criptate
- ID unico per ogni telecomando
- Blocco degli accessi non autorizzati
- Difesa contro gli attacchi predittivi o in risposta

Soluzione RAC Atmel – *la tua soluzione!*

I nostri micro ad alte prestazioni AVR con la tecnologia picoPower™ riducono il costo dei tuoi progetti. Grazie all'uso dell'algoritmo di crittografia AES, non devi più pensare a licenze, brevetti o altri costi nascosti. Anche il software è disponibile gratuitamente, così come l'ambiente di sviluppo AVR Studio®.

Codice MIP 267013

Grazie a tools di sviluppo di rinomata qualità , Atmel ti offre tutto ciò che ti serve per progettare le tue applicazioni RAC in maniera facile e flessibile

Per maggiori informazioni visita: **www.atmel.com/AVRman**

Rubrica di appuntamenti, manifestazioni ed eventi nel mondo dell'elettronica

Gli eventi di Settembre 2007

1-2 Settembre 2007

XXIXª Mostra Mercato Radiantistico

Mostra mercato dedicata all'elettronica, radiantistica, informatica, telefonia, radio, componentistica, computer, TV-Sat, editoria specializzata, strumentazione.

Informazioni utili:

Centro Fiera del Garda – Montichiari (BS)
Organizzatore Centro Fiera Del Garda
www.centrofiera.it
Orario: dalle 09.00 alle 18.30
€6,00 (ridotto €3,00)

Codice MIP 267301

8-9 Settembre 2007

34ª Mostra mercato nazionale del materiale radiantistico e delle telecomunicazioni

Mostra mercato dedicata all'elettronica, radiantistica, informatica, telefonia, radio, componentistica, computer, TV-Sat, editoria specializzata, strumentazione. Accanto al settore più propriamente "elettronico" la manifestazione presenta un'impedibile kermesse di collezioni per tutti gli appassionati.

Informazioni utili:

Quartiere Fieristico – Piacenza
Organizzatore Piacenza Expo
www.teleradio.piacenzaexpo.it
Orario: dalle 09.00 alle 18.00
€6,00 (ridotto €4,00)

Codice MIP 267302

8-9 Settembre 2007

3ª Computerfest & Radioamatore

Mostra mercato dedicata all'elettronica, radiantistica, informatica, telefonia, radio, componentistica, computer, TV-Sat, editoria specializzata, strumentazione.

Informazioni utili:

Quartiere Fieristico – Reggio Emilia
Organizzazione Compendio Fiere
www.compendiofiere.it
Orario dalle 9:00 alle 18:00

Codice MIP 267303

8-9 Settembre 2007

Mercatino scambio tra Radioamatori

Punto d'incontro tra OM è un evento riservato ai Radioamatori e organizzato da Radioamatori, come risposta al calo di materiale radioantistico nelle fiere tradizionali, che nel corso degli anni si sono trasformate in fiere di giochi, ferramenta e gadget che nulla hanno a che vedere con gli OM. Nel corso degli anni questo evento è cresciuto ed ora conta su numeri veramente importanti alla pari delle grandi manifestazioni europee.

Informazioni utili:

Caravan Camping Club – Marzaglia (MO)
Organizzatore ARI Modena
www.arimodena.it

Codice MIP 267304

15-16 Settembre 2007

Fiera Nazionale dell'Elettronica Applicata

Mostra mercato dedicata all'elettronica, radiantistica, informatica, telefonia, radio, componentistica, computer, TV-Sat, editoria specializzata, strumentazione.

Informazioni utili:

Quartiere Fieristico – Villa Potenza (MC)
Organizzazione C.B. Club Maceratese
www.cbclubmaceratese.com

Codice MIP 267305

15-16 Settembre 2007

Expo Elettronica

Elettronica nuova, usata e da collezione; collezionismo elettronico ma non solo. Expo Elettronica è la mostra mercato dedicata all'elettronica e punto

d'incontro fra "antiquariato tecnologico" e applicazioni "futuribili": una miriade di oggetti e applicazioni ormai indispensabili come computer, software, periferiche, home theater, telefonia fissa e mobile, accessori, ricambi, curiosità elettroniche e digitali. Parallelamente si svolge Colleziosa, dove si trova di tutto un po': vecchie radio, macchine fotografiche, dischi d'epoca e cd rari, fumetti, numismatica, filatelia, modellismo, giocattoli, soldatini, sorpresine, ed altro ancora, rigorosamente usato e da collezione.

Informazioni utili:

Quartiere Fieristico – Cesena

Organizzazione Blu nautilus
www.blunautilus.it

Orario dalle 9:00 alle 18:00

€7,00 (ridotto €6,00)

Codice MIP 267306

22-23 Settembre 2007

5ª Computerfest & Radioamatore

Mostra mercato dedicata all'elettronica, radiantistica, informatica, telefonia, radio, componentistica, computer, TV-Sat, editoria specializzata, strumentazione.

Informazioni utili:

Malpensa Fiere – Busto Arsizio (VA)

Organizzazione Compendio Fiere
www.compendiofiere.it

Orario dalle 9:00 alle 18:00

€ 7,00 (ridotto € 5,00)

Codice MIP 267307

22-23 Settembre 2007

Radiant And Silicon

A livello espositivo sono centinaia gli operatori che si sono avvicendati negli stand di Novegro provenienti da varie regioni del nostro Paese ed anche dall'estero costituendo un punto di grande attrazione per migliaia di appassionati. Attualmente RADIANT, che si sviluppa su una superficie espositiva superiore ai 10.000 mq. coperti, con più di 180 espositori per edizione ed oltre 40.000 visitatori annui, è considerata la più importante Mostra-Mercato nazionale del suo settore.

Informazioni utili:

Parco Esposizioni Novegro – Segrate (MI)

Orario: dalle 09:00 alle 18:00

Organizzazione: COMIS
www.parcoesposizioninovegro.it

Incontra Inware Edizioni

Codice MIP 267308

26-27 Settembre 2007

VON Italy 2007

Dopo il grande debutto del 2006, VON torna a Roma per la seconda edizione di VON Italy, l'evento italiano e mediterraneo per eccellenza sul VoIP e le IP Communications. VON ha alle spalle una storia di 10 anni e, ormai, 12 appuntamenti annuali in tutto il mondo, ed è una conferenza-esposizione che Wireless sta organizzando per conto di Pulvermedia. Organizzato su due giorni, VON Italy 2007 vedrà la partecipazione tutti i maggiori player del settore TLC, da provider di rete fissa, wireless e di broadband a fornitori di apparati.

Informazioni utili

Marriott Park Hotel – Roma

Organizzazione: Wireless S.r.l.
www.vonitaly.com

Orario: dalle 09:00 alle 18:00

Ingresso gratuito

Codice MIP 267309

27 Settembre 2007

Videogov Summit 2007

La terza tappa di Videogov Summit si terrà a Roma, il 27 Settembre. Focus sul settore dei servizi di vigilanza privata, a supporto o completamento di quanto operato dalle Istituzioni in termini di pubblica sicurezza di luoghi aperti al pubblico quali banche, centri commerciali, aree industriali, locali pubblici; e sul settore del turismo e intrattenimento, per il monitoraggio e la diffusione di immagini di località turistiche, e per la sicurezza di locali pubblici, eventi e manifestazioni.

Informazioni utili:

Marriot Park Hotel - Roma

Orario: dalle 09:00 alle 18:00

Organizzazione: Wireless srl
www.videogov.com

Codice MIP 267310

29-30 Settembre 2007

Fiera dell' Elettronica e del Radioamatore

I visitatori della fiera Millenaria, potranno trovare un vastissimo assortimento di novità e prodotti tra i quali spiccano: materiale elettronico, computers, software, cartucce, materiale radioimpiantistico, surplus, componentistica, telematica, decoder, parabole e telefonia con la partecipazione dell'editoria specializzata.

Informazioni utili:

Fiera Millenaria – Gonzaga (MN)

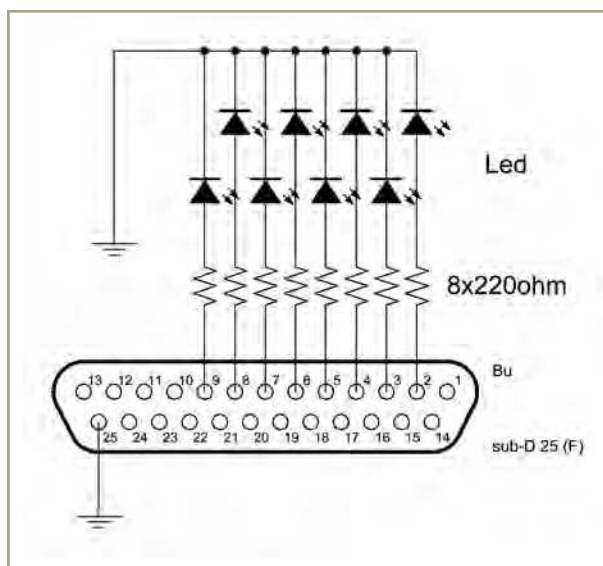
Orario: dalle 08:30 alle 18:00

Organizzazione: Fiera Millenaria
www.fieramillenaria.it

Codice MIP 267311

Dalla Redazione di Fare Elettronica una raccolta di idee ed applicazioni da tenere sempre a portata di mano.

Idee di progetto

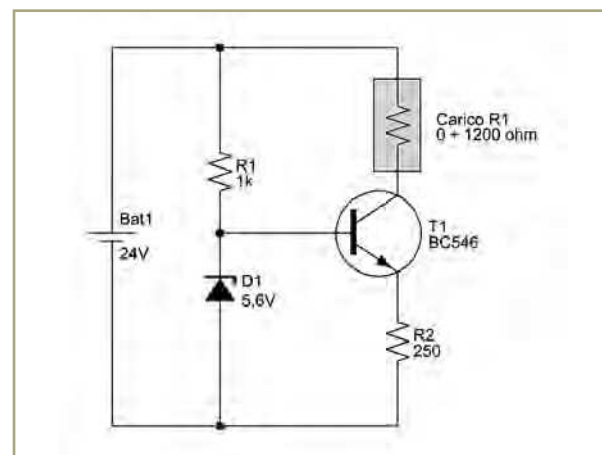
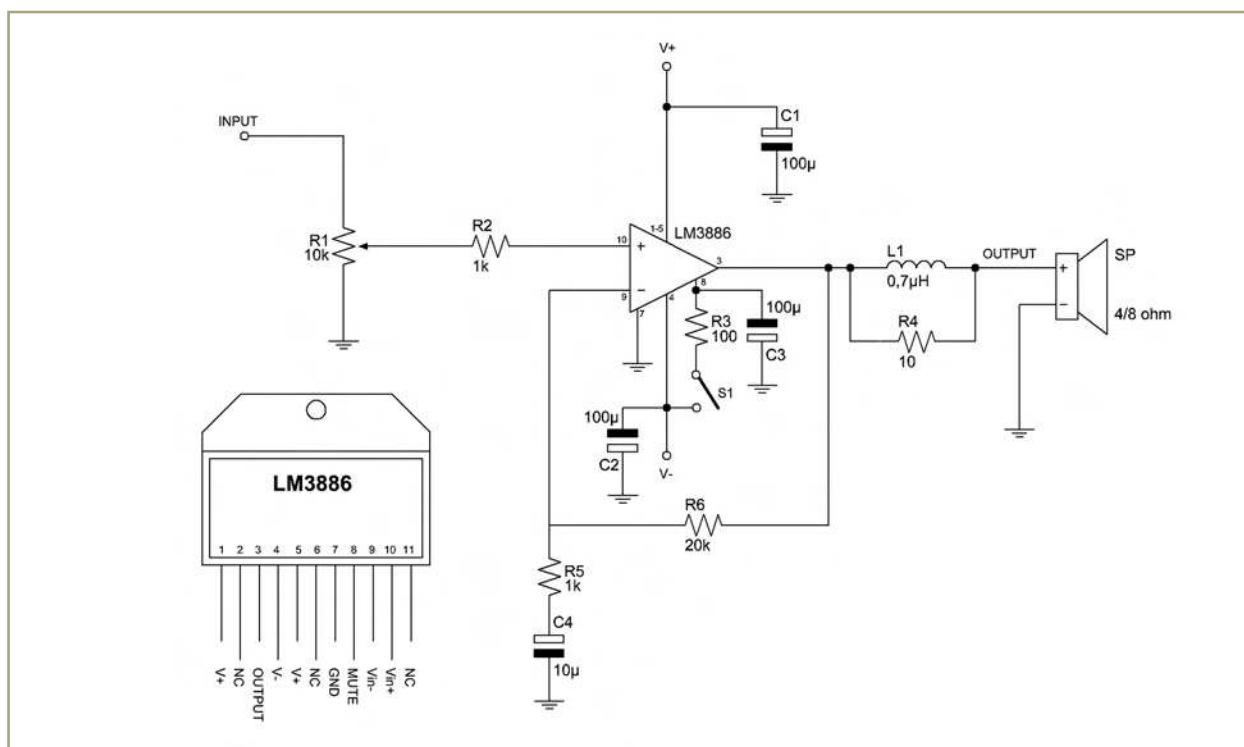
89 - Pilotare la porta parallela del PC

Con la porta parallela del Personal computer si possono pilotare fino a 8 carichi contemporaneamente. Naturalmente il carico deve poter assorbire un massimo di 20mA, poiché l'interfaccia non consente assorbimenti maggiori. L'uso di transistor e fotoaccoppiatori elimina il problema. Nello schema accanto, i carichi sono rappresentati da altrettanti diodi led, precedute dalle relative resistenze di limitazione. Il software, scritto in basic, provvede ad illuminare i diodi. È da notare che ogni pin della parallela (nel nostro caso da 2 a 9) fungono esattamente da bit di un ipotetico byte. L'esempio a seguire (scritto con il compilatore Firstbas) provvede a far lampeggiare il primo diodo, alla cadenza di un secondo. L'indirizzo della porta parallela (LPT1) è 888_{10}, ossia 378_{16}. Se si utilizza un diverso compilatore, sostituire il comando "delay" con uno equivalente.

```
While 1
   Out 888,1
   Delay 0.5
   Out 888,0
   Delay 0.5
Wend
```

90 - Generatore di corrente costante

Spesso si ha la necessità di erogare una corrente costante su un carico, qualunque sia il suo valore ohmico. Lo schema a fianco risolve brillantemente lo scopo. D1 è l'elemento stabilizzatore, che forza la base di T1 alla tensione fissa di 5,6V. L'emettitore di T1 ha pertanto una tensione costante, che assicura un passaggio di corrente costante attraverso la sua resistenza. La corrente di collettore è quindi sempre la stessa, qualunque sia il carico ad esso applicato (almento in un range compreso tra 0 e 1200 ohm). Se si volesse un diverso valore di corrente costante basta sostituire la resistenza di emettitore R2 con un altro di valore diverso, secondo la formula:

$$R_E = \frac{V_Z - 0,6}{I_{OCCORRENTE}} = \frac{5,6 - 0,6}{20mA} = 250\ ohm$$

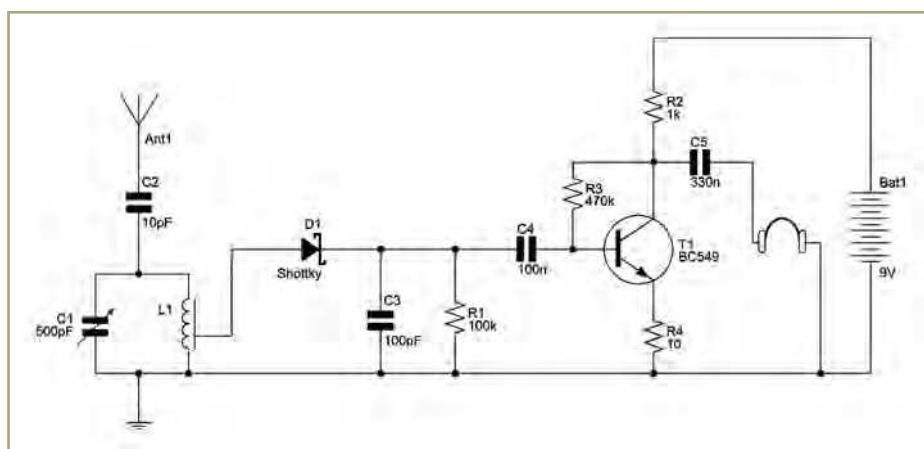
La corrente che attraversa il carico, è estremamente costante fino ad un valore di circa 1000 ohm. Dopo di che scende drasticamente. Per eliminare questo problema è sufficiente innalzare ulteriormente la tensione di alimentazione.

91 - Amplificatore audio da 68W dalle alte prestazioni

Se volete animare serate con della buona musica ad alto volume, l'integrato LM3886 fa proprio per voi. Esso è capace di fornire ben 68W di potenza audio con un carico di 4 ohm e di 38 W con un carico da 8 ohm. La distorsione armonica si attesta su circa 0,1% su tutto lo spettro acustico umano (20 Hz – 20 Khz). Le sue protezioni prevedono i casi più disparati, contro l'innalzamento immediato di temperatura, il superamento della tensione di alimentazione e della tensione del segnale, il cortocircuito e la temperatura di lavoro troppo elevata.

Per ottenere diverse potenze occorre riferirsi ai seguenti valori di carico e di alimentazione.
Si consiglia di consultare i datasheet ufficiale del componente per maggiori dettagli ed informazioni.

Potenza	Carico	Tensione di Alimentazione
68 Watt	4 ohm	±28 volt
38 Watt	8 ohm	±28 Volt
50 Watt	8 ohm	±35 Volt

92 - Radio AM d'emergenza

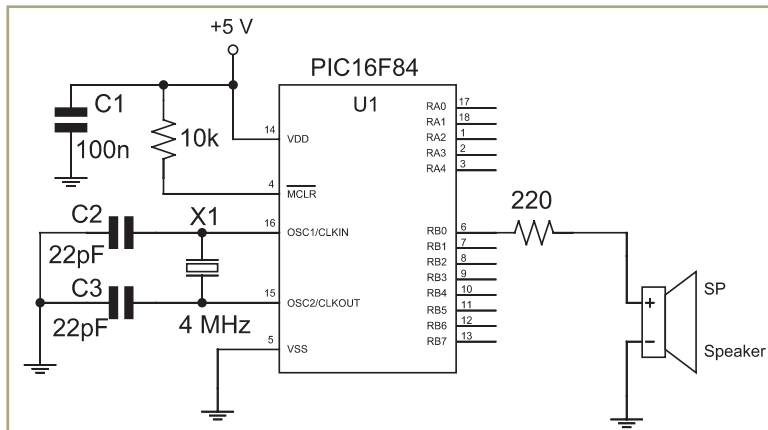
Se volete realizzare un semplice ricevitore radio che operi nella banda delle Onde Medie sulla frequenza destinata alle stazioni broadcast, compresa tra 540 Khz e 1600 Khz, allora lo schema raffigurato a lato fa per voi. L'ascolto avviene attraverso un auricolare o una cuffia ad alta impedenza (>10.000 ohm).

Il condensatore variabile da 500 pF può essere reperito sul surplus delle radio usate, mentre la bobina d'antenna è realizzata avvolgendo intorno ad un tubo di plastica o cartone circa 90 spire di filo smaltato da 5/10 con presa alla 30ma spira o, in mancanza di questa, si può utilizzare una bobina d'aereo con nucleo di ferrite. È opportuno dotare il ricevitore di un'antenna e presa di terra molto efficenti. Si consiglia l'ascolto nelle ore serali, dove la propagazione aumenta di parecchio.

93 - Divisore per n con 4017

L'integrato CD4017, si sa, è un contatore/divisore per 10 (decadico). Con una particolare configurazione si può adoperarlo quale divisore per n. Basta infatti collegare una delle uscite direttamente al piedino reset, per ottenere sul pin Q0 il segnale diviso per un fattore differente. A tale scopo si consulti la tabella qui riportata.

INTERRUTTORE CHIUSO	FATTORE DI DIVISIONE
Q2	Segnale di ingresso diviso 2
Q3	Segnale di ingresso diviso 3
Q4	Segnale di ingresso diviso 4
Q5	Segnale di ingresso diviso 5
Q6	Segnale di ingresso diviso 6
Q7	Segnale di ingresso diviso 7
Q8	Segnale di ingresso diviso 8
Q9	Segnale di ingresso diviso 9

94 - Messaggio SOS

Il famosissimo messaggio "S.O.S" codificato in codice MORSE può essere facilmente riprodotto da un microcontrollore PIC. Basta infatti implementare un semplice software che permetta l'emissione dei segnali nella giusta successione e durata, e da un microcontrollore dotato di un piccolo altoparlante. Come tutti sanno, il messaggio S.O.S (save our souls=salvate le nostre anime), è inviato dai mezzi di comunicazione in pericolo, anche se ormai, con le nuove tecnologie, tale metodologia di comunicazione sta pian piano scomparendo. Esso è composto da tre punti, tre linee e tre punti (… --- …). Il software segue le temporizzazioni utilizzate di norma, infatti la linea ha una lunghezza temporale pari a tre punti, tra un simbolo e l'altro c'è una pausa di un punto e tra una lettera e la successiva c'è una pausa di tre punti. Anche se nello schema è indicato l'utilizzo del microcontrollore Pic 16F84, si può adoperare il modello 12F675 per le dimensione molto più ridotte.

```
rem SOS Morse
rem by Giovanni Di Maria
program sos
sub procedure suono(dim limite as word)
dim k as word
for k=1 to limite
    portb.0=1
    delay_us(500)
    portb.0=0
    delay_us(500)
next k
end sub
main:
trisb=0      'PORTB in Output
portb=0      'Spegne il Display
while true
    suono(100) 'punto
    delay_ms(100)
    suono(100) 'punto
    delay_ms(100)
    suono(100) 'punto
    delay_ms(300)
    suono(300) 'linea
    delay_ms(100)
    suono(300) 'linea
    delay_ms(100)
    suono(300) 'linea
    delay_ms(300)
    suono(100) 'punto
    delay_ms(100)
    suono(100) 'punto
    delay_ms(100)
    suono(100) 'punto
    delay_ms(1000)
wend
end.
```

www.farelettronica.com

Il portale della rivista

La sezione downloads

Tutto il materiale relativo agli articoli ovvero eventuali listati sorgente e compilati, software di supporto, datasheet è disponibile nell'area downloads del portale della rivista. Ecco come trovare ciò che cercate.

Fare elettronica
IL PORTALE DELLA RIVISTA
DI ELETTRONICA APPLICATA

Utente:
Coupon: Clicca qui per ottenere un coupon utilizzabile con il tuo prossimo acquisto!

Menù
- Abbonati subito!
- Rivista
- ElettroQuiz
- Prima Pagina
- Eventi
- More Info Please!
- Forum
- IE Shop
- Area Utente
- Chi siamo
- Domande e risposte
- Contattaci

Contenuti della rivista
Richiesta copia saggio
Area download
Fare Elettronica Collection

DOWNLOAD

...e scaricare tutti i files inerenti gli articoli pubblicati sulla rivista.
...ato l'articolo associato, la dimensione, la versione e la data di ...icare il file è sufficiente cliccare sull'icona a sinistra che, tra l'altro, ...state scaricando.
...o anche riportati nella descrizione di ogni articolo, quindi, utilizzando ...ivista), visualizzate l'articolo che vi interessa e cliccate su dettagli.

Seleziona il numero di rivista 264 Vai

Sponsored Links
INWARE EDIZIONI
Expo Elettronica
elettroshop www.elettroshop.com

INWARE EDIZIONI © 2007 Inware Edizioni - Tutti i diritti sono riservati

Fare elettronica

Gli articoli della rivista sono strutturati in modo da fornire al lettore tutti gli strumenti necessari alla comprensione e alla eventuale realizzazione dell'applicazione. Spesso però l'articolo richiede materiale impossibile da pubblicare (un software, un datasheet, un programma,...) e in questo caso il portale si rivela uno strumento davvero utile. Cliccando sulla voce"Area download" dal menu "Rivista" si accede infatti ad un pagina in cui si può scegliere il numero della rivista. Scelto il numero e selezionato il pulsante "VAI" vengono mostrati tutti i downloads disponibili per quel numero. L'icona vi dà già l'informazione sul formato del file (ZIP, PDF, EXE,...) mentre le dimensioni sono riportate esplicitamente. Se state cercando i download relativi ad uno specifico articolo potete utilizzare il motore di ricerca accedendo al menu *Rivista->Contenuti della Rivista* e selezionare il numero della rivista o il titolo dell'articolo. Una volta giunti nella pagine dei dettagli dell'articolo, i relativi download saranno disponibili in fondo alla pagina. Può capitare che l'articolo comprenda anche un circuito stampato, in questo caso potrete trovare il file del PCB nella pagina dei dettagli dell'articolo sotto il nome dell'autore. Qualora disponibile, il pcb viene fornito o in formato grafico (tipicamente TIF) in scala 1:1 o in formato LAY. In quest'ultimo caso il file è visualizzabile con sPrint Layout o con l'apposito visualizzatore disponibile gratuitamente online. Ovviamente se disponete del software sPrint Layout potrete anche modificare il disegno adattandolo alle vostre esigenze. Qualora invece l'articolo preveda dei listati, questi saranno disponibili in un file zip contenente i singoli file in formato testo. Questo vi consentirà di utilizzare direttamente i file semplicemente modificandone l'estensione a seconda del compilatore oppure fare un copia/incolla del contenuto per riportarlo nell'ambiente IDE del vostro sistema di sviluppo. L'uso di del formato TXT consente di avere file di dimensioni ridotte e di eliminare tutti i caratteri speciali di formattazione introdotti dai word processor e che potrebbero causare problemi durante la fase di compilazione.

Per qualsiasi problema sul download potete contattarci all'indirizzo *mailbox@inwaredizioni.it*.

I downloads disponibili per un articolo

I downloads per la rivista n. 264

More Info Please!
Inserisci il Codice 267020 alla pagina
www.farelettronica.com/mip
oppure utilizza il **modulo** a pagina **3**

Telecamera motorizzata

Realizziamo un brandeggio motorizzato adatto a qualsiasi telecamera, ideale per sorvegliare negozi o abitazioni o per rendere le videochiamate con il vostro PC più divertenti.

Sfogliando i cataloghi professionali a proposito di CCTV (acronimo inglese di "televisione a circuito chiuso") si vede che alcuni sistemi come i brandeggi sono molto cari e quindi poco abbordabili per privati o piccole aziende, ma il sistema che vi proponiamo è estremamente conveniente in quanto viene usato un servo motore a basso costo e materiali facilmente reperibili nei negozi del fai da te. Ma prima di passare alla descrizione dettagliata del progetto facciamo una breve descrizione dei brandeggi per CCTV, del funzionamento dei servo motori e la tecnica che occorre per pilotarli.

I brandeggi per CCTV

I brandeggi sono meccanismi che permettono di fare muovere la telecamera ruotandola in alto o in basso, allo scopo di esplorare con essa una zona più ampia di quella ottenibile lasciandola puntata in una direzione. In pratica un sistema di brandeggio prevede che la staffa di fissaggio possa venire mossa da due bracci motorizzati comandati mediante segnali elettrici da un apposito pannello remoto. Questi dispositivi consentono una certa libertà di movimento, che è espressa in gradi di rotazione, intendendo con ciò l'angolo che la telecamera può compiere complessivamente in una certa direzione e non quello compiuto rispetto alla posizione di riposo. I brandeggi permettono solitamente escursioni di 355° sul piano orizzontale e 90° sul piano verticale.

Il progetto che andiamo a descrivere permette una rotazione automatica 355° sul piano orizzontale, regolabile tramite il pulsante fine corsa ed una velocità di rotazione regolabile tramite appropriato trimmer situato sulla scheda di controllo.

Il servo motore RC

Un Servo motore RC tipicamente consiste in una piccola scatola di plastica di ridotte dimensioni da cui fuoriesce un perno in grado di ruotare in un angolo compreso tra 0 e 180 gradi e mantenere stabilmente la posizione raggiunta. La rotazione del perno è ottenuta tramite un motore in corrente continua ed un meccanismo di demoltiplica che consente di ottenere un'ottima coppia in fase di rotazione. L'azionamento del motore è effettuato tramite un circuito di controllo interno in grado di rilevare l'angolo di rotazione raggiunto dal perno tramite un potenziometro resistitivo e bloccare il motore sul punto desiderato. In dotazione al servo vengono anche fornite una o più squadrette forate da poter innestare sul perno per trasmettere il movimento ad altre parti meccaniche. I Servi RC sono nati per essere pilotati nel modo più semplice possibile. L'esigenza principale era quella di poter eseguire la movimentazione senza l'ausilio di circuiterie troppo complesse. Un servo RC dispone solitamente di soli tre fili assicurati ad un connettore femmina per pin strip a passo 2,54mm: Due di questi fili sono riservati all'alimentazione in corrente continua a 5 volt. Il positivo è di colore rosso, il negativo di colore nero. L'assorbimento massimo è di circa 250mA per un Hitec HS-325HB ma può variare secondo le caratteristiche del servo. Il terzo filo, normalmente di colore giallo, è riservato per il controllo della posizione. Su questo filo è necessario applicare un segnale

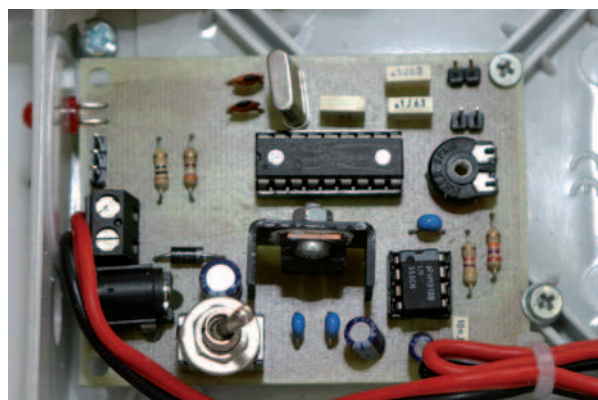



Figura 1
Foto della scheda di comando

per CCTV

di Giuseppe La Rosa

Figura 2
Foto del brandeggio con la mini telecamera

impulsivo o PWM (dall'inglese Pulse With Modulation) le cui caratteristiche sono "quasi" univoche per qualsiasi Servo RC disponibile in commercio. Per essere sicuri di riuscire a pilotare qualsiasi Servo RC il nostro circuito di pilotaggio dovrà essere in grado di inviare al servo circa 50 impulsi positivi al secondo di durata variabile in un intervallo massimo compreso tra 0,25ms e 2,75ms. Generalmente con un impulso di durata pari a 1,5ms (vedi figura 3) il perno del servo RC si pone esattamente al centro del suo intervallo di rotazione. Da questo punto il perno può ruotare fino a -90 gradi (senso antiorario) se l'impulso fornito ha una durata inferiore a 1,5ms e fino +90 gradi (senso orario) se l'impulso fornito ha durata superiore a 1,5ms. Il rapporto esatto tra la rotazione del perno e la larghezza dell'impulso fornito può variare tra i vari modelli di servo. Il servo motore usato in quest'applicazione è HS-325HB è uno dei più robusti e sicuri servi che Hitec offre ad un prezzo molto contenuto. Esso è costituito da un cuscinetto a sfere, ingranaggi ultraresistenti in Karbonite e circuiteria avanzata, l'HS-325HB offre una buona coppia unita ad un'ottima risoluzione e un centraggio preciso. Con l'utilizzo degli ingranaggi in Karbonite, l'Hitec ha risolto i problemi di durata e resistenza degli ingranaggi, questo materiale non si consuma neanche dopo migliaia d'ore d'utilizzo, garantendo una precisione eccellente nel tempo. Questi ingranaggi riescono a sopportare carichi gravosi che distruggerebbero i normali ingranaggi in nylon. Questo servo, dalle dimensioni standard, è un fuoriclasse nella sua categoria, quindi adatto ad essere utilizzato nel nostro brandeggio.

Schema elettrico

Lo schema elettrico, visibile in figura 4, è molto semplice poiché tutte le operazioni logiche e di controllo sono svolte dal microcontrollore PIC16F628 (IC1) che si occupa cosi della parte operativa del circuito. Come si può notare il circuito viene alimentato dall'ingresso X2 in cui viene applicata una tensione di 12Vcc che va ad alimentare la scheda di controllo e la telecamera. Tale ingresso è provvisto di un diodo D1 che protegge l'intero circuito dalle inversioni di polarità. Al morsetto X1 va connessa l'alimentazione della mini telecamera. Il microcontrollore IC1 ed il timer NE555 (IC3) sono alimentati con una tensione continua di +5V che è ridotta dal regolatore di tensione IC2. Gli strip JP1 e JP2 sono i pulsanti fine corsa essi servono ad

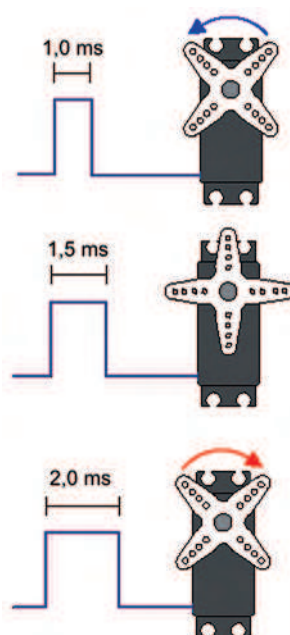



Figura 2
Relazione tra segnale PWM e rotazione del perno di un servo motore

Figura 4
Schema elettrico della scheda di controllo

ELENCO COMPONENTI

R1	Trimmer 1MΩ	LED1	Led rosso 5mm
R2	390Ω 1/4 W	S2	Interruttore a leva da C.S.
R3	27KΩ 1/4 W	X1	Morsetti a 2 poli
R4	4,7KΩ 1/4 W	X2	Spina d'alimentazione da C.S. da 2,5mm
R5	10Ω 1/4 W	JP1, JP2	Strip maschio 2 poli passo 2,54mm
C1, C2	100nF poliestere	JP2	Strip maschio 2 poli passo 2,54mm
C3, C4	22pF ceramico	JP4	Ponticello
C5, C6, C11	100nF multistrato	KK2	Dissipatore MIL7
C7	100μF 25V elettrolitico	N.1	Servo motore Hitec HS-325HB
C8	10μF 25V elettrolitico	N.1	Scatola stagna 100x100mm a pareti lisce
C9	1μF 25V elettrolitico	N.1	Staffa metallica a L 90x90x65mm
C10	10nF poliestere	N.2	Vite 15x4mm più dadi e rondelle
D1	1N4007 diodo	N.1	Spinotto maschio volante da 2,5mm
IC1	PIC16F628A	N.2	Mini pulsante passo 6,3mm
IC2	L78S05CV	N.2	Connettori molex passo 2,54mm
IC3	LM555	N.3	Vite auto filettanti 2,2x6mm
Q2	Quarzo 4Mhz	N.1	Vite auto filettanti 3,5x6mm

Figura 5
Piano di montaggio

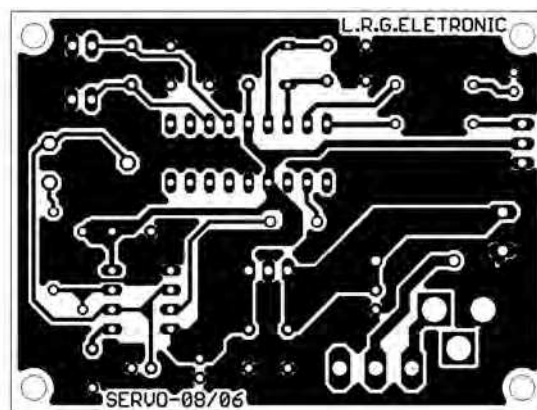



Figura 6
Circuito stampato in scala 1:1 (lato rame)

indicare al microcontrollore quando ha raggiunto l'angolo di rotazione impostato. Se JP1 è chiuso il servo motore girerà in seno antiorario, mentre se JP2 è chiuso il servo motore girerà in senso orario. Per JP1 e JP2 non abbiamo usato resistenze di pull-up esterne, perché ci siamo avvalsi dell'opzione della porta RB che permette di collegare RB7 e RB6 alle resistenze di pull-up (+Vdd) interne. Abbiamo quindi preferito inserire i soli condensatori antirimbalzo C1 e C2. Al connettore JP3 va connesso il servo motore: nei suoi contatti è disponibile la tensione d'alimentazione e il segnale di controllo PWM che è inviato dalla porta RA1 del microcontrollore IC1. La resistenza R5 posta in serie all'uscita della porta RA1, ha la funzione di abbassare l'ampiezza del segnale PWM in modo di adattarlo alle specifiche d'ingresso del servo motore. Gli impulsi PWM generati dal microcontrollore sono direttamente controllati dal timer NE555 che consente di controllarne il tempo d'invio al servo motore e di conseguenza di variarne la velocità di rotazione. L'NE555 (IC3) che lavora in questo caso da multivibratore astabile ossia è caratterizzato da due stati (alto e basso) e oscilla senza bisogno di comandi esterni. Il periodo di oscillazione è determinato dai componenti passivi C9, R1, e R3. Esso produce un segnale rettangolare con duty-cycle di poco superiore al 50%, prelevabile tra la massa e il piedino 3; la frequenza di questo segnale può variare tra circa 6Hz e 20KHz, agendo sul cursore del trimmer R1. Il diodo led (LED1) indica che il circuito è alimentato ed inoltre lampeggia alla stessa frequenza degli impulsi PWM inviati al servo

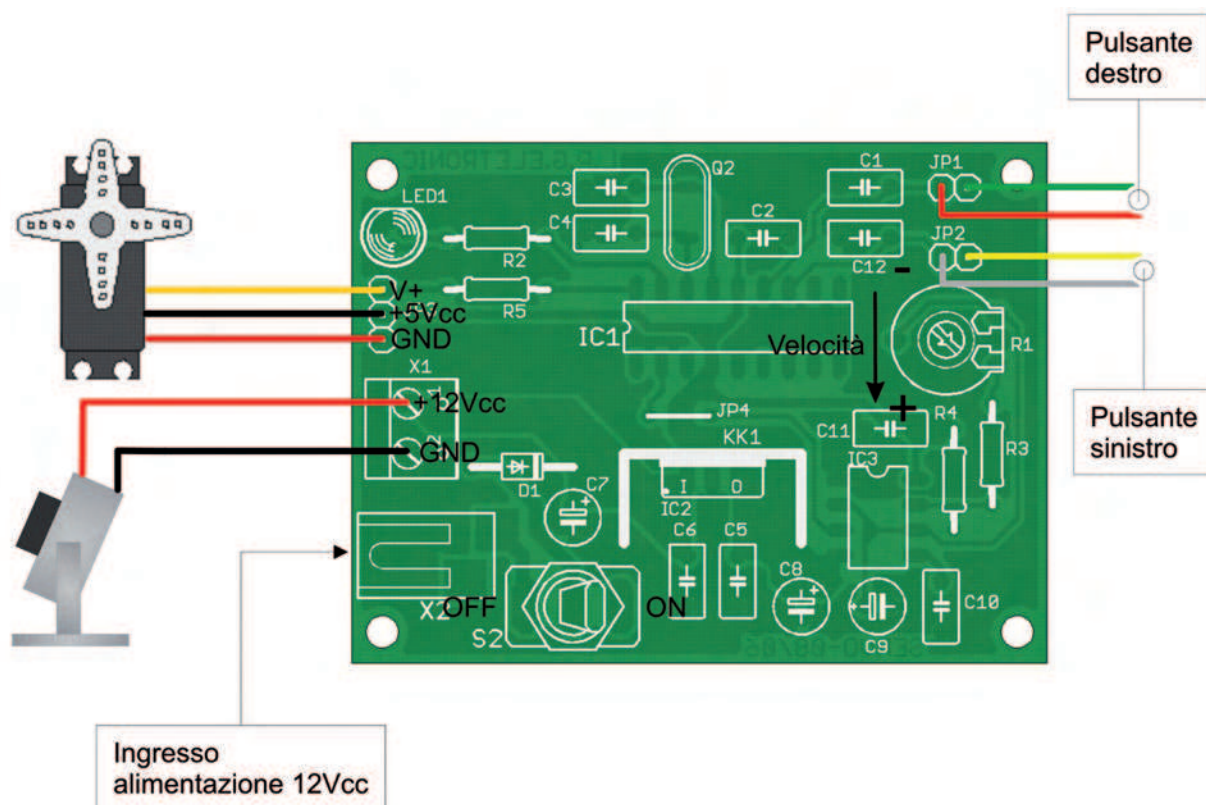



Figura 7
Schema di collegamento al servo motore ed ai pulsanti fine corsa

motore. Il valore degli impulsi PWM è salvato nella EEPROM del microcontrollore in modo da salvare la posizione del perno (del servo motore) quando non c'è alimentazione, evitando bruschi avvii del brandeggio che possono danneggiare i pulsanti fine corsa. Il sorgente e il file eseguibile sono disponibili sul sito di Fare Elettronica.

Realizzazione pratica

Il montaggio della piastra è molto semplice anche se implica molta attenzione. Occorre naturalmente realizzare il circuito stampato, la cui traccia è riportata in figura 6. Si consiglia vivamente di utilizzare la tecnica della fotoincisione o dei fogli blu. Occorre praticare i fori sulle piazzole con una punta di trapano da 0,8mm. Finita la fase di realizzazione della piastra, si può procedere al montaggio dei componenti, iniziando da quelli a basso profilo, come le resistenze il diodo D1 e i condensatori per poi passare a quelli di profilo maggiore, come lo zoccolo d'IC1 il morsetto X1 e il connettore X2. Prestate molta attenzione ai componenti polarizzati come i condensatori elettrolitici C7, C8, e C9. Usate un piccolo saldatore a punta fine, della potenza di circa di 25W. La figura 5 mostra la disposizione dei componenti sul piano d'assemblaggio. Finita la fase di assemblaggio non resta che programmare il microcontrollore. Prima di passare al montaggio della scheda nel contenitore effettuate il collaudo collegando il servo motore e l'alimentazione come illustrato in figura 7.

Assemblaggio nel contenitore

In figura 8 abbiamo riportato il disegno quotato con tutte le misure per compiere la foratura del contenitore. Il contenitore usato per questa applicazione è una scatola per derivazioni elettriche (quello usato da noi è di marca ABB) a pareti lisce dalle dimensioni 100x100mm. Iniziate la foratura dal coperchio. Ma prima tracciate tutti fori con una matita, poi passate alla realizzazione del foro per il passaggio dei cavi della telecamera e dei fine corsa, con una punta da 5mm realizzate dei fori a contornare il rettangolo per l'alloggiamento del servo motore dove l'avete tracciato in modo da poter asportare la plastica in acceso, in seguito limate i contorni del rettangolo in modo da ottenere contorni levigati e paralleli.

Attuate la stessa operazione anche per il frontale laterale della scatola prima tracciate tutti i fori con la matita e poi forate la scatola con le misure indicate in figura 8. Completata la foratura del contenitore potete fissare la staffa che servirà all'istallazione a parete del brandeggio. Usate una staffa metallica a L 90x90x65mm e fissatela con due viti 15x4mm

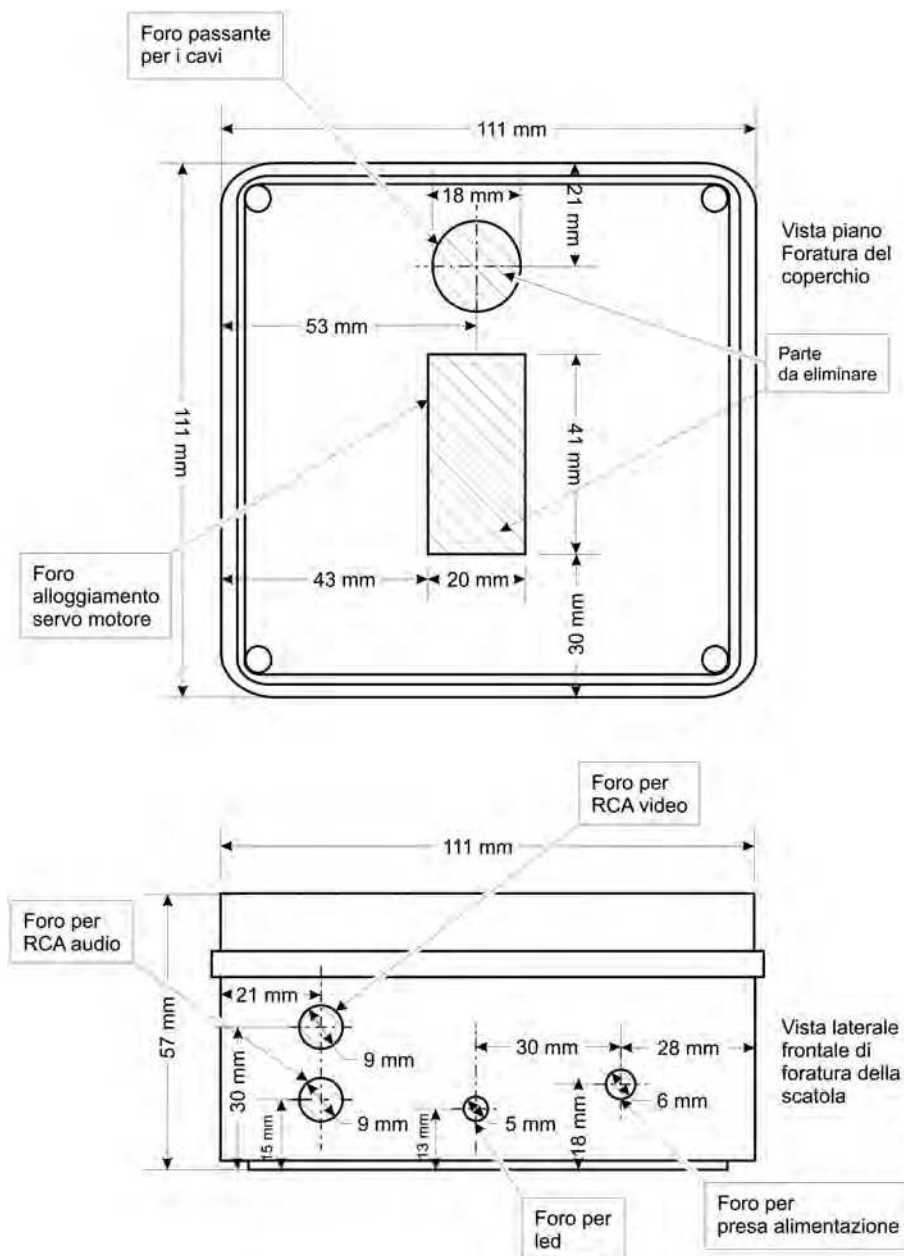



Figura 8
Piano di foratura della scatola

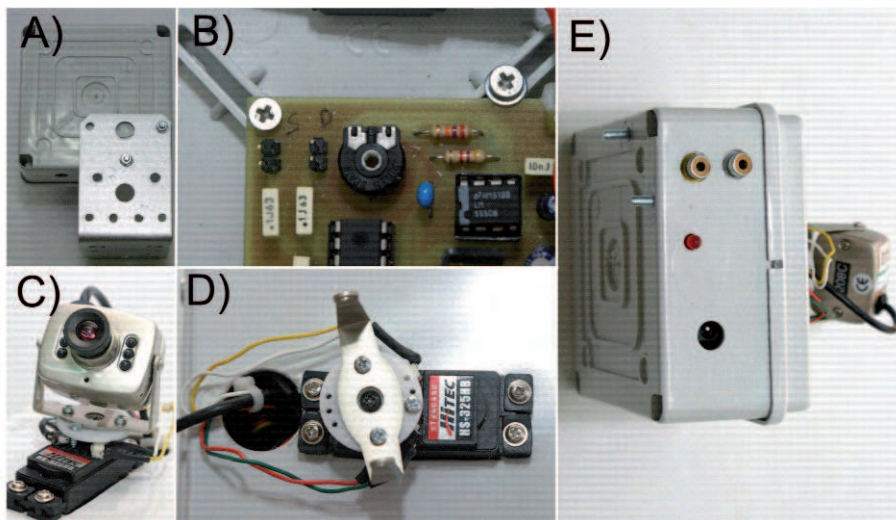



Figura 9
Vista di alcuni importati particolari di montaggio

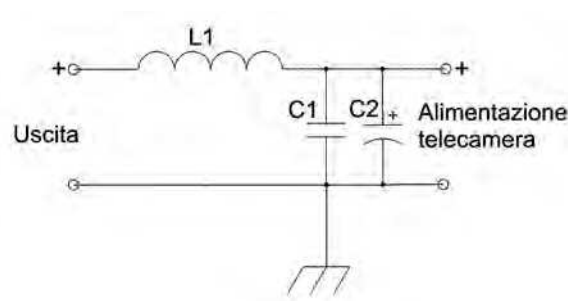



Figura 10
Eventuale filtro antidisturbo per la mini telecamera

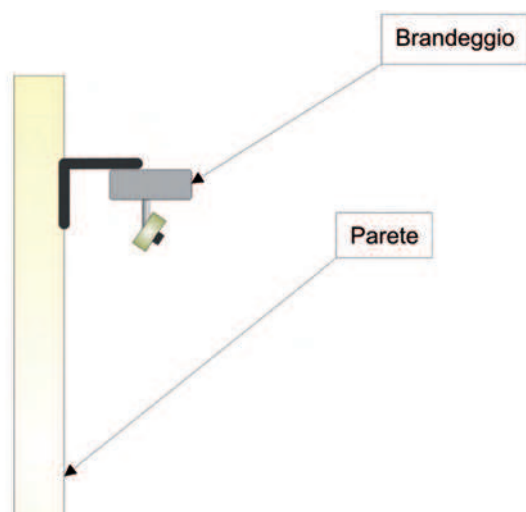



Figura 11
Esempio d'installazione a parete

come mostra la figura 9a. Passate alla collocazione della scheda di controllo nel contenitore bloccatela come illustrato nella figura 9b adoperando due viti auto filettanti 3,5x6mm.
Applicate i pulsanti fine corsa, tramite colla (super attak) al servo motore (vedi figura 9c), ponendoli con l'angolo desiderato e avvitate la vite (vite auto filettanti 2,2x6mm) al disco forato del servo, che servirà a pigiare il pulsante quando avrà raggiunto l'angolo prestabilito.
Fissate il servo motore al coperchio con le quattro viti che trovate in dotazione nella confezione del servo motore. Ora potete montare la mini telecamera sul disco forato del servo usando sempre le viti auto filettanti 2,2x6mm, come raffigurato in figura 9d. Con della colla a caldo incollate gli spinotti RCA audio e video della telecamera alla parete laterale frontale della scatola vedi figura 9e. Poi effettuate con uno spezzone di cavetto rosso e nero il collegamento tra il morsetto X1 e lo spinotto maschio da 2,5mm, che consente di alimentare la telecamera. Si conclude cosi la parte di montaggio meccanico, in figura 11 è rappresentata la tipica installazione su una parete bastano due tasselli per fissare al muro il brandeggio.

Filtro antidisturbo

Il filtro antidisturbo di figura 10 serve ad eliminare eventuali disturbi del servo motore, che passando attraverso l'alimentazione possono compromettere la stabilita dell'immagine. Da utilizzare solo in alcuni casi in cui l'immagine al monitor è particolarmente disturbata. La bobina L1 di valore 20µH corrente di lavoro 2A insieme ai condensatori C1 da 100nF e C2 di 10µF eliminano i disturbi sulla linea d'alimentazione della telecamera. Nel disegno di figura 10 é inteso con la dicitura "uscita" il morsetto X1 della scheda di controllo, e con la dicitura "alimentazione telecamera" lo spinotto maschio da 2,5mm.

Conclusione

Nella sua semplicità questo progetto, troverà il favore di chi ha necessità di tenere sotto controllo vasti ambienti, e si rivelerà utilissimo non solo nell'utilizzo come brandeggio per il quale è stato progettato, ma si presterà a tantissime altre applicazioni. Il firmware è disponibile per il download dal sito di Fare Elettronica e sicuramente, potrà essere adattato a qualsiasi utilizzo.

More Info Please!
Inserisci il Codice **267022** alla pagina
www.farelettronica.com/mip
oppure utilizza il **modulo** a pagina **3**

Network-enable

Una serie di prodotti che consentono di collegare qualsiasi periferica dotata di linea seriale ad una LAN di tipo Ethernet. Firmware aggiornabile via Internet, software disponibile gratuitamente sia per Windows che per Linus.

Nuova piattaforma EM1000

EM1000 - Ethernet Module con memoria Flash 512KB

Modulo Ethernet per la nuovissima piattaforma di casa Tibbo denominata EM1000 programmabile in Basic per lo sviluppo di applicazioni Embedded-industrial-networking. Il sistema operativo, i tools di sviluppo e il linguaggio Tibbo Basic sono disponibili gratuitamente sul sito *www.tibbo.com*. Il modulo è dotato di porta Ethernet 100BaseT e di memoria Flash da 512KB. Processore RISC (Reduced Instruction Set Computer): 88MIPS; protocolli supportati: UDP, ICMP (ping), DHCP, HTTP; fino a 16 connessioni simultanee UDP o TCP.

[EM1000-512-01 Euro 59,00]

EM1000EV - Scheda di valutazione e programmazione

Scheda di valutazione e programmazione per moduli Ethernet Tibbo appartenenti alla famiglia EM1000. Il modulo EM1000 non è incluso.

[EM1000EV • Euro 152,00]

EM1000SK - Starter Kit

[EM1000SK • Euro 255,00]

Kit di valutazione e programmazione che comprende la demoboard (EM1000EV) completa di modulo EM1000, 2 cavi UTP schermati non incrociati con connettori RJ45, 2 cavi con connettore seriale DB9 e un adattatore di rete da 12Vdc / 0,5A.

Tutti i prezzi si intendono IVA inclusa.

EM100 Ethernet Module

Realizzato appositamente per collegare qualsiasi periferica munita di porta seriale ad una LAN tramite una connessione Ethernet. Dispone di un indirizzo IP proprio facilmente impostabile tramite la LAN o la porta seriale. Questo dispositivo consente di realizzare apparecchiature "stand-alone" per numerose applicazioni in rete. Software e firmware disponibili gratuitamente.

[EM100 • Euro 52,00]

DS100 Serial Device Server

- *Convertitore completo 10BaseT/Seriale;*
- *Compatibile con il modulo EM100.*

Server di Periferiche Seriali in grado di collegare un dispositivo munito di porta seriale RS232 standard ad una LAN Ethernet, permettendo quindi l'accesso a tutti i PC della rete locale o da Internet senza dover modificare il software esistente. Dispone di un indirizzo IP ed implementa i protocolli UDP, TCP, ARP e ICMP. Alimentazione a 12V con assorbimento massimo di 150mA. Led per la segnalazione di stato e la connessione alla rete Ethernet.

[Disponibile anche nella versione con porta multistandard RS232 / RS422 / RS485, codice prodotto DS100B - Euro 134,00].

[DS100 • Euro 115,00]

EM120 Ethernet Module

Simile al modulo EM100 ma con dimensioni più contenute. L'hardware comprende una porta Ethernet 10BaseT, una porta seriale, alcune linee di I/O supplementari per impieghi generici ed un processore il cui firmware svolge le funzioni di "ponte" tra la porta Ethernet e la porta seriale. Il terminale Ethernet può essere connesso direttamente ad una presa RJ45 con filtri mentre dal lato "seriale" è possibile una connessione diretta con microcontrollori, microprocessori, UART, ecc.

[EM120 • Euro 54,00]

Via Adige, 11 • 21013 GALLARATE (VA)
Tel. 0331/799775 • Fax 0331/778112 - *www.futuranet.it*

Disponibili presso i migliori negozi di elettronica o nel nostro punto vendita di Gallarate (VA). Caratteristiche tecniche e vendita on-line: www.futuranet.it

EM200 Ethernet Module

Si differenzia dagli altri moduli Tibbo per la disponibilità di una porta Ethernet compatibile 100/10BaseT e per le ridotte dimensioni (32.1 x 18.5 x 7.3 mm). Il modulo è pin-to pin compatibile con il modello EM120 ed utilizza lo stesso software messo a punto per tutti gli altri moduli di conversione Ethernet/seriale. L'hardware non comprende i filtri magnetici per la porta Ethernet. Dispone di due buffer da 4096 byte e supporta i protocolli UDP, TCP, ARP, ICMP (PING) e DHCP.

[EM200 • Euro 58,00]

DS202R Tibbo

Ultimo dispositivo Serial Device Server nato in casa Tibbo, è perfettamente compatibile con il modello DS100 ed è caratterizzato da dimensioni estremamente compatte. Dispone di porta Ethernet 10/100BaseT, di buffer 12K*2 e di un più ampio range di alimentazione che va da 10 a 25VDC. Inoltre viene fornito con i driver per il corretto funzionamento in ambiente Windows e alcuni software di gestione e di programmazione.

È anche disponibile il kit completo comprendente oltre al Servial Device Server DS202R, l'adattatore da rete (12VDC/500mA) e 4 cavi che permettono di collegare il DS202R alla rete o ai dispositivi con interfaccia seriale o Ethernet [DS202R-KIT - Euro 144,00].

[DS202R • Euro 134,00]

EM202 Ethernet Module

Modulo di conversione Seriale/Ethernet integrato all'interno di un connettore RJ45. Particolarmente compatto, dispone di quattro led di segnalazione posti sul connettore. Uscita seriale TTL full-duplex e half-duplex con velocità di trasmissione sino a 115 Kbps. Compatibile con tutti gli altri moduli Tibbo e con i relativi software applicativi. Porta Ethernet compatibile 100/10BaseT.

[EM202 • Euro 69,00]

EM202EV Ethernet Demoboard

Scheda di valutazione per i moduli EM202 Tibbo. Questo circuito consente un rapido apprendimento delle funzionalità del modulo di conversione Ethernet/seriale EM202 (la scheda viene fornita con un modulo). Il dispositivo può essere utilizzato come un Server Device stand-alone. L'Evaluation board implementa un pulsante di setup, una seriale RS232 con connettore DB9M, i led di stato e uno stadio switching al quale può essere applicata la tensione di alimentazione (9-24VDC).

[EM202EV • Euro 102,00]

Tutti i dispositivi della serie 200 (DS202R-EM202-EM200) sono ora programmabili grazie a Taiko, una soluzione di Tibbo Technology, che vi permette di realizzare programmi che verranno eseguiti direttamente dal nuovo sistema operativo implementato nel modulo Tibbo (TiOS). Con Taiko potrete scrivere i vostri programmi direttamente in BASIC usando un semplice tool di sviluppo (TIDE) e compilarli in modo che la Virtual Machine del TiOS possa eseguirli. Questo nuovo concetto permette di trasformare un semplice "serial-to-network converters", in qualcosa di molto più sofisticato, in grado di eseguire autonomamente alcune funzioni, di filtrare dati, inviare email, creare WebServer e molto di più. I modelli mantengono le funzionalità di gateway seriale su ethernet, ma grazie a 64K di flash al loro interno, ai tool di sviluppo ed agli esempi messi a disposizione gratuitamente sul sito www.tibbo.com, i nuovi dispositivi possono essere ora programmati semplicemente dall'utente per sviluppare applicazioni dedicate.

ACCESSORI

- DMK100 - Supporto DIN per convertitori Tibbo DS100/DS202 - Euro 6,70
- TB100 - Adattatore da connettore DB9 a morsettiera per moduli Tibbo - Euro 9,50
- APR1015- Alimentatore 12Vdc / 500mA per moduli Tibbo - Euro 7,80

Codice MIP 267029

Videogames by Example

In questa puntata verrà presentato il linguaggio Spin, le sue caratteristiche ed il modo in cui può essere utilizzato per programmare il processore Propeller e la scheda si sviluppo Hydra.

Lo Spin

Lo Spin è un linguaggio ad alto livello che è stato appositamente ideato per la programmazione del Propeller. Esso presenta molte somiglianze con altri linguaggi (ad esempio Basic, C, C++, Pascal) ma ha diverse caratteristiche che lo rendono molto particolare sia nella sintassi che nell'utilizzo.
La scelta di utilizzare un linguaggio diverso da quelli esistenti è stata fatta per rendere più semplice ed efficiente la programmazione del Propeller. Questo processore infatti, come descritto nella scorsa puntata contiene al suo interno 8 processori a 32 bit che lavorano in parallelo, e che possono eseguire frammenti di codice diversi. La programmazione di un simile componente richiede quindi di gestire il codice, i parametri e le variabili relative ai singoli processori e la loro interazione. Inoltre, dal momento che in molte applicazioni alcuni processori saranno utilizzati per implementare delle periferiche particolari (es. interfacce di comunicazione), è utile avere un supporto adeguato per il riutilizzo e la modularità del codice. Da questo punto di vista lo Spin somiglia ad un linguaggio ad oggetti come il C++, nel senso che permette di definire delle entità (chiamate appunto "oggetti") che è possibile riutilizzare, e che sono in grado di funzionare autonomamente o di essere gestite da altri programmi attraverso un'interfaccia fissa. Gli oggetti possono anche essere instanziati diverse volte ed eseguiti in parallelo, come avviene con i thread del Java (con la differenza che sul Propeller l'esecuzione dei thread è realmente concorrente!).
A differenza dei linguaggi ad oggetti veri e propri comunque lo Spin è molto più semplice ed essenziale nei costrutti, al punto che il codice prodotto rispecchia da vicino le istruzioni e la struttura usate nel sorgente. Questo permette di avere una percezione precisa di cosa sta eseguendo il processore ed in che modo (similmente a quanto succede con l'assembler).
Un'ulteriore similitudine con il Java è che anche lo Spin è un linguaggio interpretato: la compilazione dei sorgenti produce un codice intermedio (byte code) che viene poi eseguito da un'interprete che viene caricato su uno dei processori presenti nel Propeller. Questa scelta rappresenta un compromesso tra l'efficienza nell'esecuzione e la compattezza del codice. Ciascuna istruzione del byte code (che è lunga 32 bit) infatti deve essere decodificata dall'interprete ed eseguita, con un sensibile dispendio di cicli macchina. Tuttavia le istruzioni del byte code traducono costrutti ed operazioni abbastanza complesse, e quindi le dimensioni del codice possono essere contenute. In effetti l'impiego principale dello Spin è quello di codificare parti dell'applicazione che inglobano complessi algoritmi di controllo, e che quindi risulterebbero più difficili da scrivere in assembler.
Quando uno dei processori è chiamato ad eseguire del codice Spin, l'interprete viene caricato sulla memoria locale del processore (il codice dell'interprete è contenuto nella ROM del Propeller) ed eseguito. L'interprete leggerà ed eseguirà il codice Spin direttamente dalla RAM principale (larga 32KB). L'uso della RAM centrale permette di avere a disposizione uno spazio maggiore per il codice, ma comporta tempi di accesso più lunghi a causa del fatto che essa è una risorsa condivisa tra i vari processori.

Struttura dei programmi

Come è stato accennato nel paragrafo precedente, lo Spin è considerato dai suoi creatori un "linguaggio ad oggetti". Questa definizione assume però un significato un po' diverso da quello classico: mentre nei linguaggi ad oggetti veri e propri è possibile definire delle classi, ciascuna dotata delle proprie variabili e funzioni ("proprietà" e "meto-

Introduzione allo Spin

di Antonio Di Stefano

di") pubbliche e privati, ed è possibile utilizzare funzionalità quali l'ereditarietà, il polimorfismo e l'overloading degli operatori, nello Spin si ha a dispone una metodologia di sviluppo e delle possibilità molto più semplici ed intuitive (sebbene altrettanto efficaci). Nello Spin in pratica gli "oggetti" sono i programmi stessi: ciascun programma infatti contiene una serie di variabili, funzioni, e dati che possono servire al suo funzionamento autonomo, ma che possono anche essere visibili dall'esterno e quindi utilizzati da altri programmi. In pratica ciascun programma può funzionare autonomamente, ma può anche essere richiamato da un altro programma, ed utilizzato come una "classe", con la possibilità di accedere ed utilizzare le sue funzioni, le variabili ed i dati.
Per rendere possibile quanto detto, i programmi hanno una struttura particolare divisa in blocchi che raccolgono tipi di informazioni simili. Questa situazione è mostrata in Figura 1, in cui è visibile la struttura di un programma Spin, ed il fatto che esso può richiamare altri programmi Spin come oggetti.

Figura 1
Struttura di un programma Spin

Il blocco CON contiene la dichiarazione di tutte le costanti utilizzate nel modulo. In pratica si tratta di semplici alias o etichette, cioè di espressioni di testo che vengono associate a dei valori fissi in modo simile a quanto avviene con le macro (#define) in C. Le costanti definite nel blocco CON, proprio perché rappresentano delle etichette sostituite dal compilatore prima della compilazione vera e propria, non occupano memoria e non devono rispecchiare tipi di dati particolari. Per questo motivo il valore delle costanti non può essere cambiato durante l'esecuzione del programma.
Un esempio di codice contenuto nel blocco CON può essere il seguente, in cui vengono definite alcune costanti relative alla grafica (il numero dei "tasselli" di una mappa e le dimensioni dello schermo):

```
CON

  ' Numero tiles orizz. e vert.
  X_TILES = 16
  Y_TILES = 12

  ' Dimensioni schermo
  SCREEN_WIDTH  = 256
  SCREEN_HEIGHT = 192
```

La sezione VAR contiene la definizione delle variabili globali utilizzate dal programma. In questo caso per ciascun elemento dichiarato viene allocato uno spazio nella memoria. In particolare le variabili globali vengono allocate nella RAM principale e non in quella locale del COG. Questo è necessario per consentire la visibilità globale e per permettere l'uso delle variabili anche mentre viene eseguito l'interprete Spin sul COG. Per ciascuna variabile è necessario specificare il tipo di dati da utilizzare. Questi sono essenzialmente il byte, la word e la long, rispettivamente a 8, 16 e 32 bit, di cui si possono instanziare anche degli array (la sintassi è la stessa del C). Le variabili globali dichiarate nella

sezione VAR (soltanto queste!) vengono automaticamente inizializzate a zero all'avvio del programma. Di seguito è riportato un esempio di blocco VAR:

```
VAR
  ' Posizione del mouse
  long mouse_x, mouse_y

  ' Mappa dei tiles
  word map[X_TILES*Y_TILES]

  ' Tavolozza dei colori
  long colors[64]

  ' Contatore piccolo
  byte cnt
```

Il blocco OBJ comprende la lista dei moduli esterni importati, che si intende utilizzare nel programma. I moduli esterni devono essere dei normali programmi Spin che verranno utilizzati come "oggetti": sarà quindi possibile richiamarne le funzioni pubbliche, le variabili ed i dati. Da questo particolare si deduce la principale differenza rispetto a linguaggi tradizionali come il C. Anche nel C è possibile richiamare moduli esterni, ma questi devono essere scritti in modo opportuno, non possono essere dei programmi in grado di funzionare anche autonomamente. In un progetto scritto in C infatti deve esserci soltanto una funzione main, che sarà contenuta soltanto in un modulo, e che verrà eseguita per prima. Nei programmi Spin invece non è presente un funzione speciale che ha proprietà rispetto alle altre, le funzioni vengono eseguite secondo il loro ordine. Grazie a questo particolare uno stesso programma può essere utilizzato indifferentemente anche come "libreria". Dal momento che ciascun modulo può richiamare altri moduli, per evitare ripetizioni il compilatore provvede automaticamente ad allocare una sola istanza di ciascun modulo richiamato. Uno degli usi tipici della sezione OBJ è quella di richiamare i driver per la gestione delle periferiche. Questo permette di riutilizzare efficientemente il codice già scritto. Di seguito è mostrato un esempio della sintassi del blocco OBJ:

```
OBJ
  ' Driver per segnale video
  tv  : "tv.spin"

  ' Gestione per la grafica
  gr :  "graphics.spin"

  ' Driver mouse
  mouse:  "mouse_io.spin"
```

La sezione PUB è associata alle funzioni pubbliche, cioè quelle funzioni che si desidera siano visibili anche dall'esterno del modulo (cioè che possano essere esportate ed utilizzate quando il modulo viene richiamato come oggetto da un altro modulo). Come già detto non esiste una funzione speciale che viene eseguita per prima, tutte le funzioni sono uguali, ma l'esecuzione inizierà dalla prima, in ordine di scrittura, del modulo "top level" (cioè il modulo che richiama gli altri e non è richiamato da nessuno).
Un esempio di sezione PUB è il seguente:

```
PUB Square (x) : risultato
  Risultato := (x*x)
```

Questa funzione semplicemente eleva al quadrato l'argomento passato (x). La sintassi delle funzioni verrà descritta più in dettaglio in seguito.
Il blocco PRI è identico a PUB, con l'unica differenza che le funzioni sono visibili ed utilizzabili soltanto dall'interno del modulo.
L'ultima sezione presente nei file Spin è il blocco DAT. In questa sezione possono essere collocati dati di qualsiasi tipo che verranno memorizzati nella RAM principale e potranno essere utilizzati dai programmi: costanti, valori, coefficienti, stringhe, tabelle, etc. I blocchi DAT sono anche utilizzati per contenere il codice assembler. L'assembler infatti è trattato come se fosse un insieme di dati binari grezzi (ed in effetti è questo dopo la compilazione). Il blocco di dati assembler, dopo essere stato generato dal compilatore, viene memorizzato nella RAM, e potrà essere caricato nella memoria di uno dei COG durante l'esecuzione del programma. Per questo motivo le istruzioni assembler devono essere meno di 512 (la memoria dei COG), a meno di non implementare particolari routine di caricamento dinamico.
Un esempio di sezione DAT è riportato di seguito:

```
DAT
  punti     word  0
  livello   byte  0
  stringa1  byte  "Game Over", 0
  stringa2  byte  "Player 1", 0
  stringa3  byte  "Player 2", 0

  ' Assembler
  org
  inizio    mov x, par
            add x, 5
            shl x, #2
            …
```

Ciascun dato presente nel blocco è identificato da una etichetta che verrà associata dal compilatore

alla locazione di memoria in cui il dato si trova (o da cui inizia). Le etichette quindi sono in pratica dei puntatori. Si può notare che per definire le stringhe si è utilizzata una sequenza di byte. Lo Spin in effetti non ha dei tipi predefiniti per le stringhe, ma utilizza lo stesso metodo del C: una sequenza di byte che termina con il valore 0. Anche l'indirizzo di inizio del codice assembler è indicato da un'etichetta. La sintassi e l'utilizzo dell'assembler verranno trattati in una puntata successiva.

Sintassi dello Spin

Gli esempi riportati in precedenza permettono di notare alcuni dettagli sulla sintassi dello Spin: il linguaggio non è case sensitive (le lettere maiuscole non sono considerate diverse dalle minuscole), non sono utilizzati dei caratteri particolari per terminare le linee (come i punti e virgola per il C) e i commenti sono preceduti da un'apice. Per commentare più linee simultaneamente è possibile racchiuderle tra parentesi graffe singole ("{" e "}") o doppie. In quest'ultimo caso i commenti saranno utilizzati per la documentazione automatica del codice.

Una particolarità dello Spin è che non utilizza caratteri o parole chiave (come parentesi o begin/end) per racchiudere i blocchi subordinati a particolari istruzioni o annidati, ma soltanto spazi. Un blocco è considerato annidato se è scritto più a destra di quello a cui fa capo. Questa particolarità evita di dovere aprire e chiudere esplicitamente i blocchi, ma può portare ad errori, tra l'altro poco visibili, in caso di distrazione. Per facilitare questo compito l'ambiente di sviluppo può indicare graficamente l'estensione dei blocchi (premendo Ctrl+I), in questo modo tra l'altro si ottiene anche un codice sempre ordinato ed ben indentato.

I tipi principali, come già accennato sono i byte, le word e le long, che è possibile utilizzare in maniera molto flessibile e diretta, con gli stessi meccanismi tipici del C (array, puntatori, stringhe, etc). Per gestire correttamente questi tipi di dati ed i relativi puntatori, è opportuno ricordare che il Propeller, e quindi anche i COG, utilizzano un'organizzazione di tipo "little endian" (la stessa dei processori Intel), cioè per i tipi che impiegano più byte, l'indirizzo più basso corrisponde al byte meno significativo, come mostrato in Figura 2.

Principali costrutti

Lo Spin ha relativamente poche istruzioni, come il C, ma relativamente più complesse e versatili. Questo permette di scrivere semplici programmi utilizzando in pratica soltanto i costrutti condizionali e quelli per l'implementazione di cicli. I costrutti condizionali sono l'if ed il case. Entrambi somigliano a quelli del C, ma hanno delle possibilità aggiuntive che risultano estremamente comode. Il costrutto if ad esempio ha la seguente sintassi:

```
if condizione1
  codice1
elseif condizione2
  codice2
ifnot condizione3
  codice3
elseifnot condizione4
  codice4
else
  codice5
```

L'unica sezione obbligatoria è la prima (quella dell'if vero e proprio), le altre sono opzionali. La sezioni elseif permette di specificare una condizione alternativa da testare rispetto a quelle precedenti, la sezione ifnot viene eseguita se la condizione specificata viene trovata falsa, la elseifnot viene eseguita se tutte le condizioni precedenti erano false ed anche la condizione specificata è falsa. La sezione else viene eseguita se nessun'altra condizione è stata verificata, quindi costituisce il caso di default. Le varie sezioni possono essere combinate in qualsiasi modo, è solo necessario mantenere l'if all'inizio e l'else (quando presente) come ultima condi-

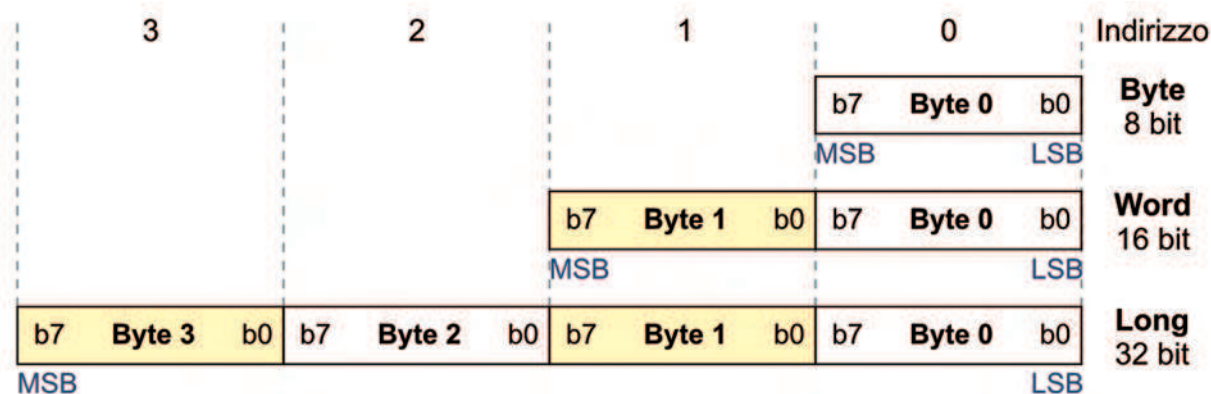



Figura 2
Organizzazione dei tipi di dati dello Spin

zione. Va notato ancora una volta che il codice che si riferisce a ciascun blocco non è racchiuso da nessun simbolo particolare che lo delimita, si riconosce solamente per il fatto di essere scritto leggermente più a sinistra della condizione a cui si riferisce. Se questa regola non viene rispettata si possono avere errori o un comportamento diverso da quello voluto.
L'altra istruzione condizionale è il case, che permette di implementare una scelta ad n vie in base al valore (numerico) di una variabile. La sintassi è molto simile a quella del C, ed è mostrata di seguito:

```
case variabile
  costante1:
    codice1
  costante2.. costante3, costante4:
    codice2_4
  costante5, costante6:
    codice5_6
  other:
    codicex
```

Le varie scelte possono essere individuate da valori singoli (come costante1), da un intervallo, indicato con i due punti "..", da un elenco, indicato con la virgola, o in generale da una combinazione di questi elementi. L'ultima condizione è quella di default indicata con other. Anche in questo caso l'indentazione del codice è una parte fondamentale della sintassi.
Il costrutto per realizzare cicli è il repeat, che può essere utilizzato in diversi modi combinandolo con altre parole chiave quali while, until, from, to, step, next e quit. Nella versione più semplice il repeat

OPERATORE	UNARIO	DESCRIZIONE
:=		Assegnazione
->		Rotazione a destra
<-		Rotazione a sinistra
>>		Shift a destra
<<		Shift a sinistra
#>		Limite minimo, es.: "x #>= y" equivale a "if x<y then x=y"
#<		Limite massimo, es.: "x #<= y" equivale a "if x>y then x=y"
-	x	Negazione
!	x	Complemento dei bit
&		AND binario
|		OR binario
^		XOR binario
| |	x	Valore assoluto
+		Somma
-		Differenza
~>		Shift aritmetico a destra
><		Inverte l'ordine degli y bit inferiori, il resto è posto a 0
AND		AND logico
OR		OR logico
NOT		NOT logico
*		Moltiplicazione con segno 32x32 bit, restituisce i 32 bit inferiori del risultato a 64bit
**		Moltiplicazione con segno 32x32 bit, restituisce i 32 bit superiori del risultato a 64bit
/		Divisione, restituisce il quoziente
//		Divisione, restituisce il resto (modulo)
^^	x	Radice quadrata
<		Operatore di confronto: minore
>		Operatore di confronto: maggiore
<>		Operatore di confronto: diverso
==		Operatore di confronto: uguale
=<		Operatore di confronto: minore o uguale
=>		Operatore di confronto: maggiore o uguale

Tabella 1
Operatori dello Spin

esegue continuamente un blocco di codice (individuato grazie all'indentazione) all'infinito, o fino a quando non viene richiamata l'istruzione next (passa al ciclo successivo) o quit (esce dal ciclo).
La sintassi è la seguente:

```
repeat n
  codice
  ' il codice può contenere
  ' condizioni con next o quit
```

L'espressione n può essere presente o meno, e può essere un'espressione numerica qualsiasi, una variabile o il valore restituito da una funzione.
Se viene utilizzata il loop sarà ripetuto esattamente n volte.
Per ripetere l'esecuzione fino a quando una condizione viene verificata o non è più verificata, si aggiungono le istruzioni while e until, che possono essere inserite all'inizio del blocco o alla fine a seconda se si desidera che il codice venga eseguito almeno una volta oppure no:

```
repeat {while/until condizione}
  codice
  ' il codice può contenere
  ' condizioni con next o quit
{while/until condizione}
```

Il costrutto repeat può anche essere utilizzato per implementare loop con variabili di conteggio (ossia come un ciclo "for"), in questo caso la sintassi è la seguente:

```
repeat contatore from inizio to fine
  codice
```

All'espressione può essere aggiunta l'istruzione step che permette di specificare gli incrementi da dare alla variabile di conteggio (come negli altri linguaggi normalmente l'incremento è 1). Inoltre nel blocco di codice possono sempre essere usate le istruzioni next e quit per influenzare l'esecuzione.

Operatori e funzioni

Uno degli aspetti più interessanti dello Spin è la sua ricchezza di operatori. Come si può vedere dalla Tabella 1 sono presenti sia i comuni operatori aritmetici, logici e relazionali, assieme ad altri più sofisticati, che permettono di eseguire in un'unica istruzione operazioni relativamente complesse. Oltre a questo sono presenti anche alcuni "modificatori" di variabili estremamente versatili, che sono riportati in Tabella 2. Tra i più singolari si possono notare quelli per "randomizzare" le variabili scorrendole tramite un LFSR (cfr. Firmware n.10), quelli per l'estenzione del segno, per l'azzeramento o l'impostazione ad 1 dei bit dopo l'uso della variabile, e quelli per l'incremento e decremento simili a quelli del C.

Le funzioni sono una parte importante dei programmi Spin, sia perché rendono più modulare e leggibile il codice, come avviene negli altri linguaggi, sia perché le funzioni, in particolare quelle pubbliche, rappresentano anche i "metodi" di ciascun oggetto, quando questo viene richiamato da un altro programma.
La sintassi delle funzioni prevede la dichiarazione del tipo (pubblica o privata), l'elenco dei parametri, dell'eventuale valore di ritorno, e le variabili locali. Per ciascuno di questi elementi non è necessario dichiarare il tipo dal momento che vengono utilizzati sempre dati a 32 bit. Un esempio di dichiarazione di funzione è il seguente:

```
PRI Distanza(x,y) : d | temp1, temp2
  codice
  return d
```

In questo caso è stato dichiarata la funzione "Distanza" che è di tipo privato, che prende in ingresso due parametri, restituisce il valore calcolato (d) ed impiega internamente due variabili locali (temp1 e temp2). Va notato che occorre rispettare gli spazi riportati tra i vari elementi della dichiarazione. Per restituire il valore, oltre ad utilizzare l'istruzione return, è anche possibile assegnare il valore alla variabile "super-globale" result. Questa variabile predefinita, che ha visibilità in qualunque parte del programma ed in tutti i moduli, contiene sempre l'ultimo valore restituito, quindi può essere anche usata direttamente. Ad esempio, per calcolare il quadrato di un numero si potrebbe scrivere:

Modificatore	**Descrizione**
?var	Scorrimento in avanti con LFSR a 32 bit, 4 tap
var?	Scorrimento indietro con LFSR a 32 bit, 4 tap
~var	Estensione del segno da variabile a 7 bit
~~var	Estensione del segno da variabile a 15 bit
var~	Post-clear a 0 (prima lettura, poi azzeramento)
var~~	Post-set a 1 (prima lettura, poi impostazione a 1)
var++	Post-incremento
++var	Pre-incremento
var–	Post-decremento
–var	Pre-decremento

Tabella 2
Modificatori di variabile dello Spin

```
PUB Quadrato(x) | temp
  temp := x*x
  result := temp
```

È interessante notare che lo Spin ha delle regole di "scoping" (visibilità delle variabili) piuttosto curiose: le variabili locali non possono avere lo stesso nome di variabili globali, ma possono avere lo stesso nome di altre variabili dichiarate in altre funzioni. La stessa regola si applica ai nomi dei parametri ed a quello dei valori di ritorno.

Esempio

Per mostrare quanto esposto nei precedenti paragrafi, è riportato di seguito un breve programma scritto in Spin "puro" che ha lo scopo di far variare gradualmente l'intensità luminosa del LED di debug presente sulla scheda Hydra. Per fare questo viene utilizzato una forma molto semplice di modulazione PWM, realizzata via software.

```
CON
' Debug LED su I/O P0
DEBUG_LED = $00000001
VAR
  ' Conteggio PWM e ritardo
  byte pwmcnt, dcnt
  ' Livello PWM e direzione
  byte level, dir
PUB lampeggio
' Inizializzazione
DIRA[0] := DEBUG_LED
level := 0
dir := 1

' Ciclo infinito
repeat while TRUE
  ' Incrementa contatori
  pwmcnt++
  dcnt++
  ' Generazione PWM
  if pwmcnt > level
    OUTA[0]:=0
  else
    OUTA[0]:=DEBUG_LED

  ' Azzeramento contatore
  if pwmcnt>15
    pwmcnt:=0
  ' Gestione livello
  if dcnt==200
    level:=level+dir
    if level==15 or level==0
      dir:=-dir
    dcnt:=0
```

Per realizzare la modulazione PWM viene utilizzato un contatore (pwmcnt) che viene fatto contare da 0 a 15 (quindi si tratta di un PWM a 4 bit). Il valore del contatore viene confrontato con quello del livello stabilito (level) ed in base al risultato si accende il LED o meno. La regolazione dell'intensità avviene facendo variare il valore del livello ogni 200 iterazioni, grazie ad un altro contatore di ritardo (dcnt). Ogni volta che il valore del livello raggiunge il massimo valore o il minimo, viene modificato il segno dell'incremento (dir), in modo da ottenere una variazione a dente di sega.

Si può notare che nel codice non è stato necessario includere nessuna sezione PRI, OBJ o DAT.

Nella prossima puntata, verranno analizzati alcuni esempi decisamente più complessi ed interessanti che coinvolgeranno anche la generazione di un segnale video e l'interfacciamento di periferiche di input.

More Info Please!
Inserisci il Codice **267030** alla pagina
www.farelettronica.com/mip
oppure utilizza il **modulo** a pagina **3**

INWARE EDIZIONI SHOP

CD-ROM

Annate complete FE su CD-ROM

Annate disponibili dal 2003 al 2006

Le annate complete in formato PDF. Potrai sfogliare comodamente tutte le riviste e stampare (senza perdere in qualità) gli articoli di tuo interesse. Ogni CD-ROM contiene anche software, codice sorgente, circuiti stampati e tutto il materiale necessario per la realizzazione dei progetti proposti. In ogni CD è presente una sezione con contenuti speciali.

COD. **FE-CD2003 € 25,80**
COD. **FE-CD2004 € 30,00**
COD. **FE-CD2005 € 30,00**
COD. **FE-CD2006 € 30,00**

PIC Microcontroller By Example

Il corso completo PIC® Microcontroller By Example in formato PDF

Tutte le lezioni pronte per la consultazione con i sorgenti dei progetti immediatamente utilizzabili nelle tue applicazioni. Il modo migliore per avere sempre sottomano la soluzione per il tuo progetto con i **PICmicro®**. Il **CD-ROM PIC® Microcontroller By Example** contiene una sezione **"Contenuti Speciali"** tutta da scoprire.

COD. **FE-PBE € 15,90**

Tutto sulle Smartcard

La raccolta completa degli articoli sulle smartcard in formato PDF

Gli articoli, i progetti e i download relativi agli articoli sulle Smartcard in un unico CD-ROM pronto da consultare ed utilizzare. Contiene i progetti completi del lettore di smartcard **UniReader** e del **SimCopier** per duplicare il contenuto della rubrica della vostra Sim card.

COD. **FE-SMARTCARD € 15,90**

Annata 2006 Firmware completa su CD-ROM

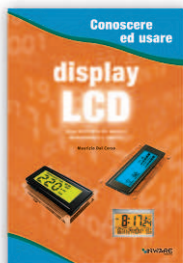
10 numeri di Firmware su un CD-ROM

Questo CD contiene tutti i numeri di Firmware usciti nel 2006 in formato PDF ad alta risoluzione. Sono presenti tutti i listati dei codici presentati sulla rivista, potrai quindi comodamente utilizzarli insieme agli articoli di tuo interesse. Il Cd contiene inoltre una sezione con interessanti contenuti speciali.

COD. **FW-CD2006 € 30,00**

Scopri i bundle e le offerte che ti abbiamo riservato

PROPOSTE 2007

Display LCD

Questo libro di successo (oltre 2000 copie vendute) rappresenta una delle migliori guide all'utilizzo dei moduli alfanumerici basati sul controller HD44780, moduli grafici con controller KS0108 e non solo. Il testo tratta anche i display LED a sette segmenti e i display LCD passivi. Numerosi gli esempi pratici di impiego dei vari dispositivi: dal contatore a 7 segmenti al termometro LCD fino al pilotaggio dei moduli alfanumerici mediante PICmicro e PC.

COD. FE-06 **€ 16,50**

PICmicro™

La lettura di questo libro è consigliata per conoscere a fondo i PICmicro seguendo un percorso estremamente pratico e stimolante. Il testo descrive l'uso di MPLAB®, e descrive, in maniera approfondita, tutte le istruzioni assembler e molte delle direttive del compilatore. Al testo è allegato un utilissimo CDROM che, oltre ai sorgenti e gli schemi dei progetti presentati nel testo, contiene moltissimi programmi di utilità e molta documentazione.

COD. FE-18 **€ 29,00 (contiene CD-ROM)**

Linguaggio ANSI C

Questo nuovissimo libro descrive le tecniche, gli accorgimenti migliori per sfruttare gli aspetti di "alto e basso livello" del C, entrambi fondamentali quando si vuole sviluppare del *firmware* per sistemi dotati di risorse limitate.
Il testo è particolarmente indicato sia a chi ha già esperienza nella programmazione in *assembler* di sistemi a microcontrollore ed intende iniziare ad utilizzare il linguaggio C, sia per chi conosce già il C e vuole avvicinarsi alla programmazione dei sistemi *embedded*.

COD. FE-25 **€ 24,90**

BASIC per PIC

Un volume indispensabile sia per chi si avvicina alla programmazione dei PIC utilizzando il linguaggio Basic, sia per chi intende affinare le proprie tecniche di programmazione. Una guida alla programmazione embedded utilizzando MikroBASIC, uno dei più completi compilatori per PIC dotato di ambiente IDE e moltissime funzioni di libreria. La trattazione vi guiderà dalla semplice accensione di un LED alla gestione di motori in PWM, alla lettura e scrittura di memorie I2C, alla generazione di suoni seguendo un percorso semplice e ricchissimo di esempi pratici.

COD. FE-27 **€ 24,90**

CPLD

Un libro dedicato a tutti coloro che per la prima volta si avvicinano al mondo delle Logiche Programmabili ed utilizzabile da quanti, già esperti, desiderano approfondire la conoscenza di questi interessanti dispositivi. Gli argomenti teorici sono presentati attraverso semplici circuiti di esempio il cui codice viene descritto nei dettagli. Tra gli argomenti trattati: la sintassi del linguaggio Verilog, la comunicazione seriale, la conversione analogico-digitale e le macchine a stati finiti.

COD. FE-28 **€ 32,90 (contiene DVD-ROM)**

Amplificatori operazionali

Un testo per capire a fondo l'amplificatore operazionale. Le tipologie, le configurazioni circuitali, l'analisi approfondita dei parametri caratteristici sono solo alcuni degli argomenti trattati nel testo. I numerosi esempi pratici fanno di questo libro un utilissimo volume adatto anche sia a chi intende conoscere gli amplificatori operazionali senza per questo affrontare lunghe trattazioni teoriche, sia per gli studenti che hanno la necessità di conoscere a fondo questo affascinante ed utilissimo componente elettronico.

COD. FE-29 **€ 39,00**

LIBRI

www.ieshop.it

Codice MIP 267039

I prezzi indicati sono IVA compresa

Settima parte: *Settembre 2007*
I Relays e la gestione dei dati

Ottava parte: *Ottobre 2007*
Utilizzare il modulo Led a 7 segmenti giganti

Nona parte: *Novembre 2007*
Utilizzare il display LCD

Il PLC per tutti

Il linguaggio Ladder è in grado di gestire in maniera molto efficiente anche i dati in formato Word e Double Word e non solo i contatti che possono assumere esclusivamente i valori logici 0 e 1. In questo articolo verrà illustrata una gestione approfondita degli I/O ed una tecnica software di gestione dei dati.

Il registro F

Come accennato la scorsa puntata, è presente, tra i vari Relay, uno speciale registro, chiamato appunto registro F. La sua funzione è quella di monitorare lo stato interno di sistema del Cubloc. Tramite i suoi Relay, infatti possiamo eseguire alcune automazioni molto utili.
La tabella 1 mostra le funzioni svolte dal registro F, con i Relays utilizzabili dall'utente finale. Riveste particolare importanza il Relay F2, che fornisce un impulso positivo, della durata di uno Scan Time, all'accensione del dispositivo, o meglio, all'esecuzione dell'istruzione "Set Ladder On".
Come si vede inoltre, vi sono due Relays che posseggono uno stato sempre stabile, ossia forzato a 0 logico (F0) oppure a 1 logico (F1). Sono molto utili quando si desidera impostare ad una tensione fissa i dispositivi. Se ad esempio occorre che un diodo Led, collegato alla porta P0, resti sempre acceso, basterà seguire il sorgente Ladder proposto in figura 1. Il Relay F1 fornisce infatti sempre un valore di 1 logico.

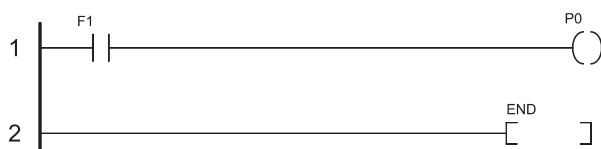



Figura 1
Un diodo Led sempre acceso

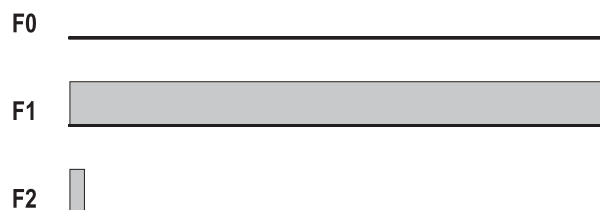



Figura 2
Diagramma di livelli dei Relays F0, F1 e F2

Un semplice lampeggiatore

Vediamo adesso come realizzare un semplice lampeggiatore. Guardando la tabella 1 ci accorgiamo che alcuni Relays (e precisamente F16 ÷ F39) forniscono un segnale di commutazione automatica ON-OFF ogni periodo prefissato. Il Relay F24 esegue ad esempio una commutazione ogni 10ms mentre il Relay F39 ogni 10 minuti circa.

Schema elettrico

Lo schema è molto semplice, prevede infatti un solo diodo Led collegato alla porta P0 del Cubloc,

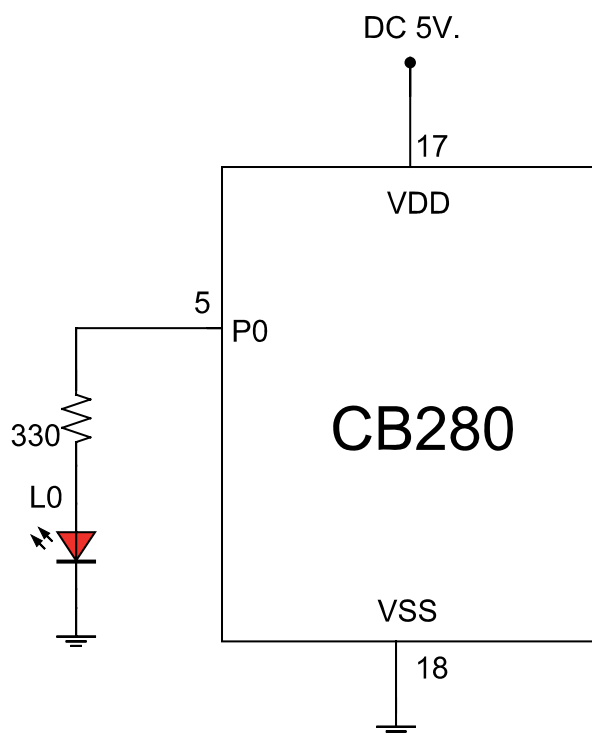



Figura 3
Schema elettrico del lampeggiatore

I Relays e la gestione dei dati

di Giovanni Di Maria

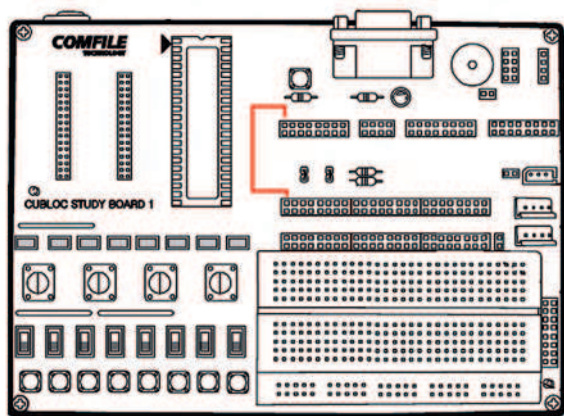



Figura 4
Disposizione dei collegamenti per il lampeggiatore

configurata come uscita. Esso è visibile in figura 3.

Sorgente Basic
Veramente semplice, prevede e configura un'unica porta di output alla quale è assegnato l'alias "led".

```
Const Device = CB280
Usepin 0,Out,led
Set Ladder On
Do
Loop
```

Sorgente Ladder
Ancor più semplice il sorgente Ladder che prevede, in un unico Rung, il lampeggio. Come si nota infatti

F0	Relay sempre in OFF	**F22**	Ripete ON/OFF ogni 64 scan time
F1	Relay sempre in ON	**F23**	Ripete ON/OFF ogni 128 scan time
F2	Level 1 Scan time at Power Up	**F24**	Ripete ON/OFF ogni 10ms
F3		**F25**	Ripete ON/OFF ogni 20ms
F4		**F26**	Ripete ON/OFF ogni 40ms
F5		**F27**	Ripete ON/OFF ogni 80ms
F6		**F28**	Ripete ON/OFF ogni 160ms
F7		**F29**	Ripete ON/OFF ogni 320ms
F8	1 Scan On ogni 10ms	**F30**	Ripete ON/OFF ogni 640ms
F9	1 Scan On ogni 100ms	**F31**	Ripete ON/OFF ogni 1.28sec.
F10		**F32**	Ripete ON/OFF ogni 2.56sec.
F11		**F33**	Ripete ON/OFF ogni 5.12sec.
F12		**F34**	Ripete ON/OFF ogni 10.24sec.
F13		**F35**	Ripete ON/OFF ogni 20.48sec.
F14		**F36**	Ripete ON/OFF ogni 40.96sec.
F15		**F37**	Ripete ON/OFF ogni 81.92sec.
F16	Ripete ON/OFF ogni 1 scan time	**F38**	Ripete ON/OFF ogni 163.84sec.
F17	Ripete ON/OFF ogni 2 scan time	**F39**	Ripete ON/OFF ogni 327.68sec.
F18	Ripete ON/OFF ogni 4 scan time	**F40**	Call Ladderint
F19	Ripete ON/OFF ogni 8 scan time	**F41**	
F20	Ripete ON/OFF ogni 16 scan time	**F42**	
F21	Ripete ON/OFF ogni 32 scan time		

Tabella 1
Le funzioni del registro F

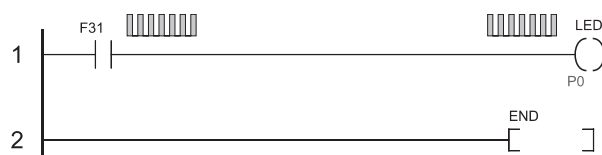



Figura 5
Sorgente Ladder del lampeggiatore

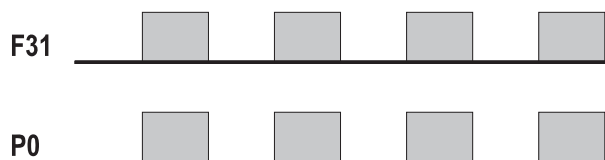



Figura 6
Diagramma di livelli del lampeggiatore

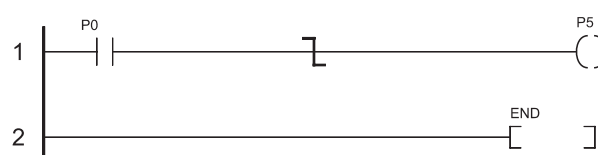



Figura 9
Esempio pratico del comando DIFD

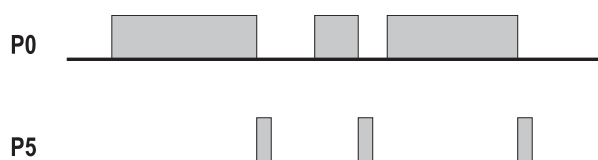



Figura 10
Diagramma di livelli del comando DIFD

in figura 5, il Relay F31, che autonomamente causa un processo di ON/OFF ogni 1,28 secondi, pilota direttamente il Relay P0 che, di conseguenza, causa il lampeggio alla frequenza specificata.

Il comando DIFU

Questo comando, integrabile in una linea di Rung, ha la funzione di fornire in uscita un breve impulso positivo quando il suo ingresso passa da uno stato logico basso ad uno alto, indipendentemente dalla durata dell'input. La sua azione avviene pertanto sul fronte di salita del segnale. Utile per fornire brevi impulsi (della durata di 1 scan time) anche quando i segnali che li hanno generato sono molto prolungati. Il simbolo è "disegnato" premendo il tasto F11. La figura 7 mostra un esempio pratico, mentre la figura 8 realizza l'andamento temporale del segnale mediante un diagramma di livello.

Il comando DIFD

Questo comando, integrabile in una linea di Rung, ha la funzione di fornire in uscita un breve impulso positivo quando il suo ingresso passa da uno stato logico alto ad uno basso, indipendentemente dalla durata dello stesso. La sua azione avviene pertanto sul fronte di discesa del segnale. Il simbolo è "disegnato" premendo il tasto F12. La figura 9 mostra un esempio pratico, mentre la figura 10 realizza l'andamento temporale del segnale mediante un diagramma di livello.

Il Ladder gestisce anche i Dati

Il linguaggio Ladder prevede non solo i contatti che possono assumere esclusivamente i valori logici 0 e 1. È infatti in grado di gestire in maniera molto efficiente anche i dati, in formato Word (16 bit) e Double Word (32 bit). Il registro che permette la loro implementazione è chiamato registro D, ed accetta 100 dati (Relays D0 ÷ D99).
Come si vede dalla mappa di memoria di figura 11, ogni cella di memoria rappresenta una word (16 bit). Un valore di tipo Double Word occuperà naturalmente due celle adiacenti. Trattandosi di Ram ovviamente è prevedibile che i dati vadano persi allo spegnimento del modulo.

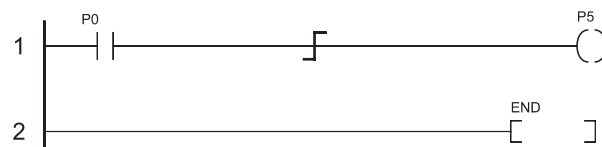



Figura 7
Esempio pratico del comando DIFU

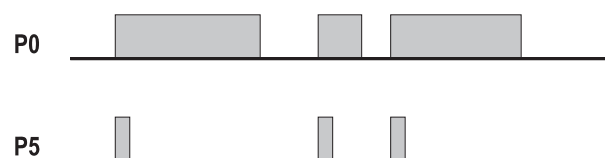



Figura 8
Diagramma di livelli del comando DIFU

Figura 11
La mappa di memoria del registro D

COMFILE
TECHNOLOGY

CB220
Controllore industriale impiegato in applicazioni e progetti che necessitano un microcontrollore programmabile o un PLC.
Il CB220 può controllare e monitorare interruttori, motori, timers, sensory, relé, valvole e molti altri dispositivi.
Il Cubloc basic ladder logic è il linguaggio usato per la programmazione. CUBLOC BASIC è simile ad altri basic presenti sul mercato e il LADDER LOGIC si avvicina agli standard PLC.

€ 50,40

CB280
Controllore industriale impiegato in applicazioni e progetti che necessitano un microcontrollore programmabile o un PLC.
Il CB280 può controllare e monitorare interruttori, motori, timers, sensory, relé, valvole e molti altri dispositivi.
Il Cubloc basic ladder logic è il linguaggio usato per la programmazione. CUBLOC BASIC è simile ad altri basic presenti sul mercato e il LADDER LOGIC si avvicina agli standard PLC.

€ 63,60

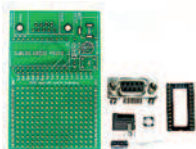
CB405
Controllore industriale impiegato in applicazioni e progetti che necessitano un microcontrollore programmabile o un PLC.
Il CB405 può controllare e monitorare interruttori, motori, timers, sensory, relé, valvole e molti altri dispositivi.
Il Cubloc basic ladder logic è il linguaggio usato per la programmazione. CUBLOC BASIC è simile ad altri basic presenti sul mercato e il LADDER LOGIC si avvicina agli standard PLC.

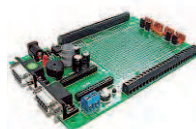
€ 78,00

CB290
Controllore industriale impiegato in applicazioni e progetti che necessitano un microcontrollore programmabile o un PLC.
Il CB220 può controllare e monitorare interruttori, motori, timers, sensory, relé, valvole e molti altri dispositivi.
Il Cubloc basic ladder logic è il linguaggio usato per la programmazione. CUBLOC BASIC è simile ad altri basic presenti sul mercato e il LADDER LOGIC si avvicina agli standard PLC.

€ 102,00

Study Board
Banco di studio e test per imparare ad usare rapidamente e facilmente i controllori Cubloc CB220 o CB280.
Grazie a svariate periferiche come LED, RS232, breadboard, pulsanti, interruttori ed altro, l'utente è in grado di usare e testare le funzionalità che il controllore offre.

€ 102,00

CB220 ProtoBoard
Kit per montare una semplce scheda (73x48 mm) per interfacciare il modulo Cubloc CB220 tramite porta seriale.
Sono inclusi tutti i componenti necessari ed è richiesta la saldatura.

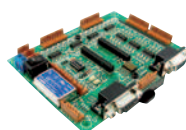
€ 7,38

CB280 ProtoBoard
Scheda per interfacciare facilmente il modulo Cubloc CB280 con linee di I/O senza creare un nuovo circuito stampato.
Con l'aggiunta di una breadboard, la scheda si può trasformare in una banco per test e sviluppo.

€ 71,40

Quick Start Board 1000
Scheda di studio e sperimentazione per controllore CB405.
Grazie a svariate periferiche come Led, ADC, switch, pulsanti, piezo, breadboard ed altro, l'utente è in grado di usare e testare le funzionalità che il controllore offre.

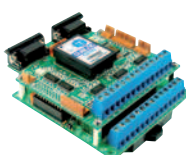
€ 71,40

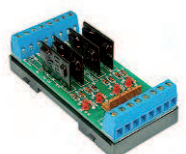
CB290 ProtoBoard
Scheda per interfacciare facilmente il modulo Cubloc CB290 con linee di I/O senza creare un nuovo circuito stampato.

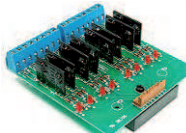
€ 102,00

CuBASE Board-32M
Controller board per Cubloc CB280 che predispone l'interfacciamento del modulo con numerose I/O come le porte PWM, 2 porte seriali, uscite di transistor NPN, AD ecc ecc.

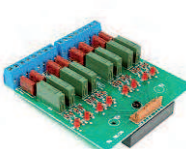
€ 114,00

CuBASE Board-64M
Controller board per Cubloc CB290 che predispone l'interfacciamento del modulo con numerose I/O come le porte PWM, 2 porte seriali, uscite di transistor NPN, AD ecc ecc.

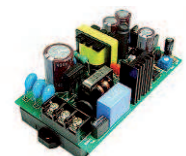
€ 186,00

CuSB-22D
Sistema integrato per il controllo industriale che comprende:
- Cubloc CB280
- Scheda periferiche
- Scheda di alimentazione 24V
- Scheda a relè

€ 166,80

SSR4 Board
Scheda con 4 relè a bordo per espandere le funzionalità del controllore Cubloc.
- Tensione in ingresso: 4~32VDC
- Alimentazione: AC50~240V
- Assorbimento corrente : 0~2A
- Dimensioni: (89 x 42 x 25mm).

€ 28,26

SSR8 Board
Scheda con 8 relè a bordo per espandere le funzionalità del controllore Cubloc.
- Tensione in ingresso: 4~32VDC
- Alimentazione: AC50~240V
- Assorbimento corrente : 0~2A

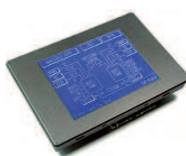
€ 58,02

Relay8 Board
Scheda con 8 relè a bordo per espandere le funzionalità del controllore Cubloc.

- Interfacciamento Plug-N-Play con Cubloc e Cutouch
- ZNR per il filtraggio del rumore
- Attacco DIN-RAIL

€ 43,14

DP17-24
Alimentatore: 85V~264V in ingresso, 24V (0.7A) in uscita
- Input : AC 85V ~ 264V
- Output : DC 24V / 0.7A (17W)
- Attacco DIN-RAIL
- Dimensioni: 89mm x 51mm X 36mm

€ 34,20

CT1720
Il kit CT1720 unisce in un unico prodotto un controllore Cubloc, un PLC e un interfaccia touch screen.
Il Cutouch trova il suo impiego in tutte quelle applicazioni che necessitano di un microcontrollore programmabile o di un PLC.
Rimpiazza il vecchio metodo di collegare un display al PLC avendo già tutto integrato.

€ 442,80

CT1721
Il kit CT1721 unisce in un unico prodotto un controllore Cubloc, un PLC e un interfaccia touch screen.
Il Cutouch trova il suo impiego in tutte quelle applicazioni che necessitano di un microcontrollore programmabile o di un PLC.
Rimpiazza il vecchio metodo di collegare un display al PLC avendo già tutto integrato.

€ 478,80

PREZZI IVA INCLUSA

MICRO PLC PROGRAMMABILI IN BASIC E IN LADDER LOGIC

Ordina i prodotti COMFILE su www.ieshop.it oppure telefona allo 02.66504755

Codice MIP 267043

I comandi per gestire i Dati

In tabella 2 sono elencati i relativi comandi supportati dal Ladder e loro utilizzo. Naturalmente non li esamineremo tutti ma vedremo semplicemente i più utilizzati, con degli esempi.

Incremento e decremento

Come esempio pratico scriviamo un programma che provvede ad incrementare o decrementare il valore di una variabile, premendo uno dei due tasti adibiti allo scopo.

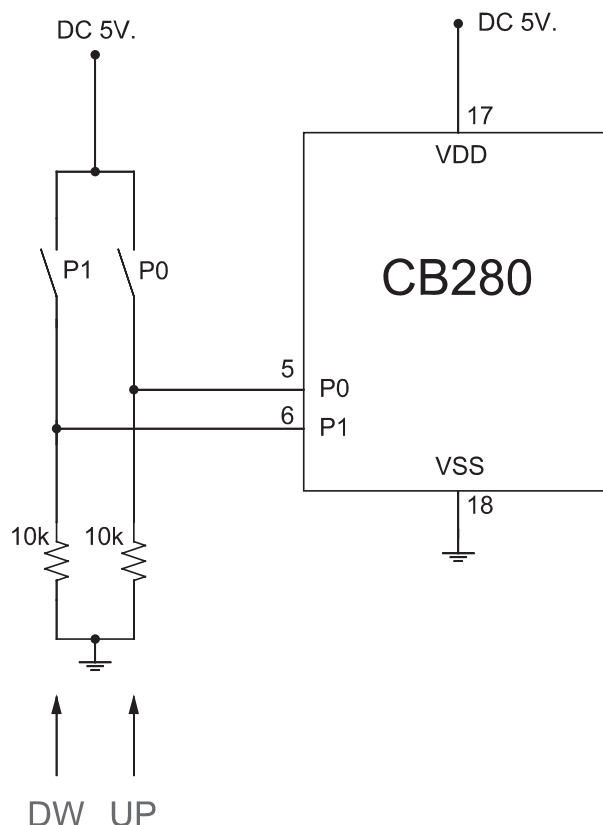



Figura 12
Schema elettrico per l'incremento e decremento

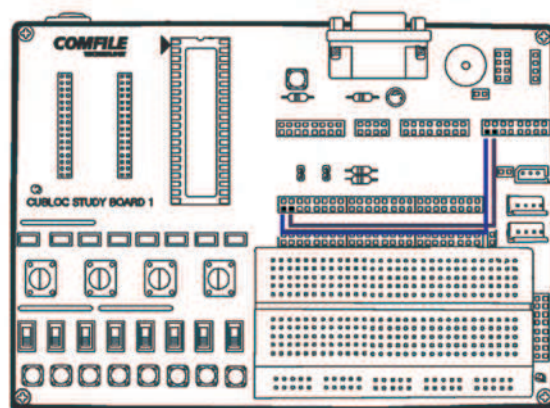



Figura 13
Disposizione dei collegamenti per l'incremento e il decremento

Schema elettrico

Lo schema prevede solo due pulsanti, collegati ad altrettante porte, configurate come ingresso. Non sono presenti porte in uscita, in quanto il risultato sarà visibile direttamente via software. Esso è visibile in figura 12, mentre in figura 13 sono illustrate le connessioni da effettuare sulla Study Board.

Sorgente Basic

Molto breve, esso provvede semplicemente a configurare due porte in ingresso e a nominarle rispettivamente pulsante1 e pulsante2. Si ricorda che l'uso dell'alias non è obbligatorio.

```
Const Device = CB280
Usepin 0,In,pulsante1
Usepin 1,In,pulsante2
Set Ladder On
```

Comando a 16 bit	Comando a 32 bit	Funzione svolta
WMOV	DWMOV	Memorizza l'operando in memoria
WXCHG	DWXCHG	Scambia il contenuto di due operandi
FMOV		Riempie le celle di memoria desiderate
GMOV		Riempie le celle di memoria desiderate
WINC	DWINC	Incrementa di un'unità l'operando
WDEC	DWDEC	Decrementa di un'unità l'operando
WADD	DWADD	Somma due operandi e memorizza in memoria
WSUB	DWSUB	Sottrae due operandi e memorizza in memoria
WMUL	DWMUL	Moltiplica due operandi e memorizza in memoria
WDIV	DWDIV	Divide due operandi e memorizza in memoria
WOR	DWOR	Effettua OR logico tra due operandi e memorizza in memoria
WXOR	DWXOR	Effettua XOR logico tra due operandi e memorizza in memoria
WAND	DWAND	Effettua AND logico tra due operandi e memorizza in memoria
WROL	DWROL	Effettua uno scorrimento di bit a sinistra (moltiplica per 2)
WROR	DWROR	Effettua uno scorrimento di bit a destra (divide per 2)

Tabella 2
I comandi Ladder per gestire i dati

```
Do
Loop
```

Sorgente Ladder

Molto più interessante il sorgente Ladder, composto da quattro Rung, di cui l'ultimo è quello finale.

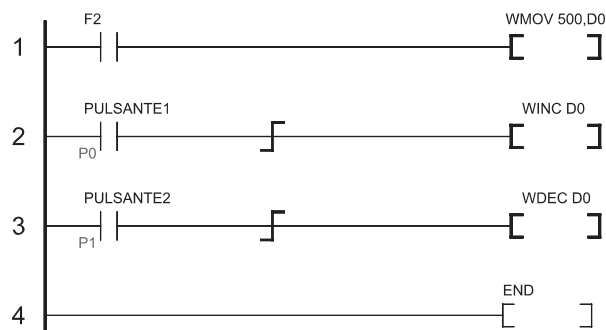



Figura 14
Sorgente Ladder dell'incremento e decremento

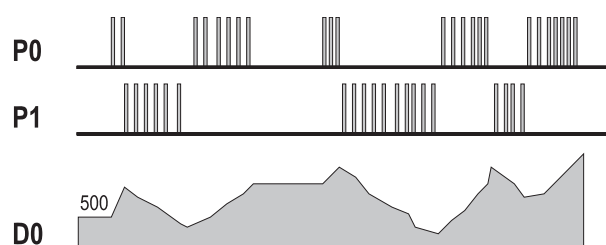



Figura 15
Diagramma di livelli dell'incremento e decremento

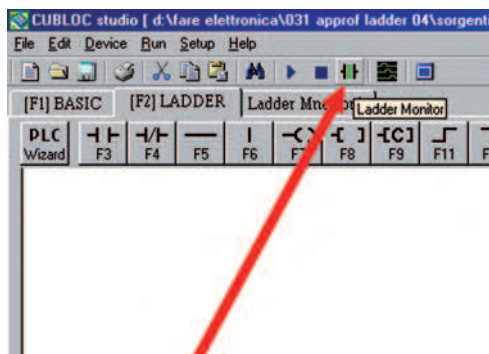



Figura 16
Il pulsante per l'attivazione del Ladder Monitor

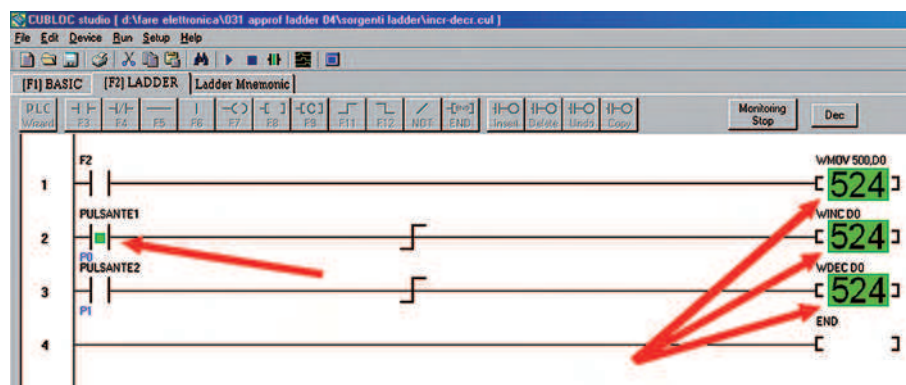



Figura 17
Il Ladder Monitor in funzione

Di seguito viene descritto il funzionamento del programma con particolare attenzione al comportamento di ogni Rung.

Il primo Rung possiede un Relay di input e uno di output. Il simbolo F2 fornisce un breve impulso positivo all'inizio dell'esecuzione, ed in questo modo permette l'attivazione del comando "WMOV 500,D0", che serve appunto a memorizzare, solo per la prima volta, il valore 500 nel registro D0. Il secondo Rung controlla lo stato del primo pulsante e, in caso di sua attivazione (valore logico 1), manda un impulso al successivo Relay, programmato con il comando "WINC D0". Esso si occupa dell'incremento di un'unità del registro D0. Il terzo Rung controlla lo stato del secondo pulsante e, in caso di sua attivazione (valore logico 1), manda un impulso al successivo Relay, programmato con il comando "WDEC D0", decrementando così di un'unità il valore del registro D0. Da notare l'utilizzo del simbolo DIFU nella seconda e terza linea, che serve "forzare" un incremento alla volta, grazie all'invio di un breve impulso sul fronte di salita del segnale generato dai pulsanti. Senza di esso, l'incremento seguirebbe un andamento molto più ampio (veloce ed incontrollato), per via della lunghezza temporale della pressione sui tasti.

Verifica

Dal momento che nell'esempio non sono state previste porte di output, l'unico modo per verificare il valore del registro D0 è quello di "spiarvi" dentro, utilizzando il "Ladder Monitor". Con esso è possibile controllare costantemente, ed in tempo reale, il valore di tutti i Relays utilizzati nel progetto. Dopo la compilazione e lo scaricamento del Firmware, a tale scopo si prema il simbolo indicato in figura 16, dalla barra degli strumenti.

In questo modo il programma sarà eseguito non solo sul Cubloc, ma anche simulato all'interno del Personal Computer, dando all'utente la possibilità di controllo su tutti i parametri coinvolti. Se a video un Relay assume il colore verde, significa che il suo valore logico è alto.

Progettiamo un semaforo

E finalmente ecco la ciliegina sulla torta. Il classico esempio, sempre preso in considerazione, che riesce a focalizzare, nella sua interezza, l'algoritmo comprensivo di tante problematiche ed esigenze pratiche e didattiche. Il prototipo è sempre trattato quale

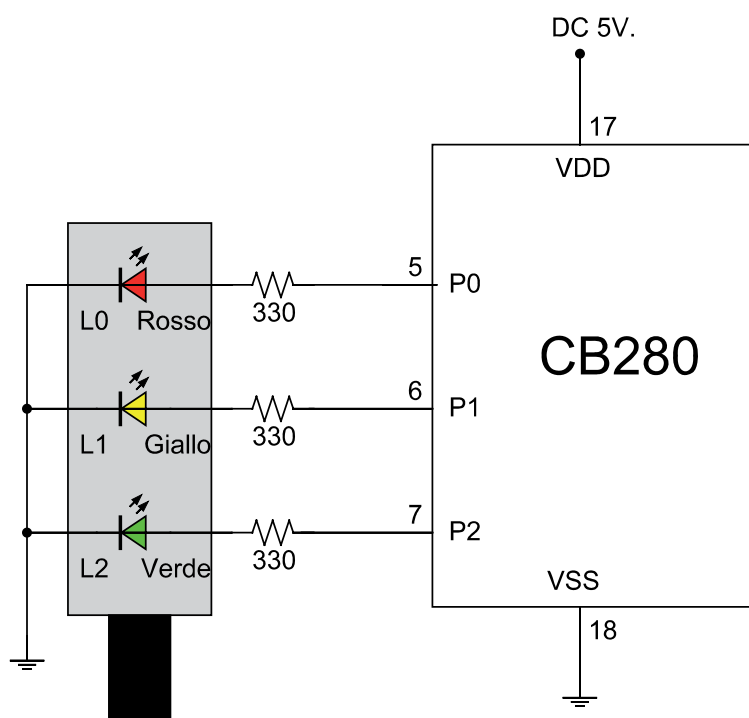



Figura 18
Schema elettrico del semaforo

esempio nelle applicazioni automatizzate.
Probabilmente l'implementazione di un semaforo in linguaggio Ladder potrebbe risultare un tantino più difficoltosa rispetto ad altri linguaggi standard (Basic o C), ma il comprendere come i simboli e Relays interagiscano tra loro, al fine di formare un progetto funzionante ed autonomo, fornisce sicuramente un bagaglio culturale senza dubbio rilevante.

Schema elettrico

Lo schema elettrico, visibile in figura 18, prevede solo tre diodi Led collegati ad altrettante porte, configurate tutte come uscite. Le porte P0, P1 e P2 del Cubloc hanno la funzione di output e pilotano direttamente i diodi indicatori, e precisamente P0 simula il rosso del semaforo, P1 il giallo e P2 il verde. In figura 19 sono illustrate le connessioni da effettuare sulla Study Board.

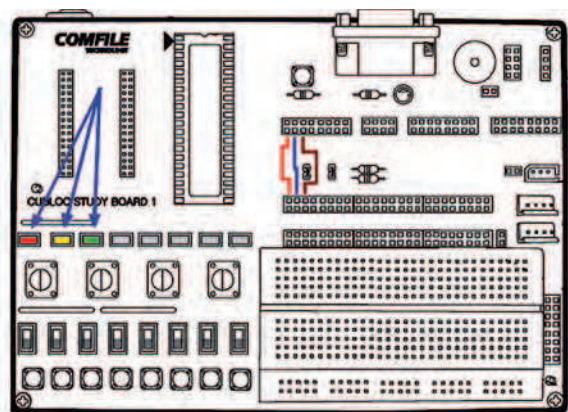



Figura 19
Disposizione dei collegamenti per il progetto semaforo

Sorgente Basic

Come al solito esso programma la funzione delle porte. L'utilizzo degli alias semplifica di molto la gestione e l'organizzazione del progetto, cosicché il programmatore non ha bisogno di ricordare il nome "fisico" della porta ma semplicemente il nome "logico" e pratico della stessa.

```
Const Device = CB280

Usepin 0,Out,rosso
Usepin 1,Out,giallo
Usepin 2,Out,verde

Set Ladder On
Do
Loop
```

Le fasi logiche del semaforo

Come avviene per tutti i prototipi automatizzati, anche il semaforo segue una sorta di "robotizzazione" logica, per le fasi operative di funzionamento. In particolare, i tre momenti esecutivi sono rappresentati dalle seguenti situazioni di lavoro:

1. Lampada VERDE accesa per 10 secondi;
2. Lampada VERDE + lampada GIALLA accese per 3 secondi;
3. Lampada ROSSA accesa per 12 secondi;
4. Si ripete la sequenza all'infinito.

Si noti che la lampada verde resta accesa per un totale di 13 secondi (10 sec + 3 sec). Naturalmente la temporizzazione può essere regolata a piacimento, secondo le esigenze.

Sorgente Ladder

Andiamo ad esaminare adesso le specifiche del sorgente Ladder che, a dire la verità, non sono così immediate e semplici. Studiate pertanto molto attentamente questa fase che è la più delicata.
Il sorgente si compone di ben 11 Rung, compreso quello finale. Come si nota, a destra dei Rung "critici" sono presenti anche dei piccoli semafori che indicano lo stato delle lampade, nei vari momenti. Andiamo a commentare le linee singolarmente.

1° Rung

Ha lo scopo di illuminare, tramite l'impulso iniziale ed il comando SETOUT, il diodo Led relativo alla luce Verde. Da notare che tale clausola è processata solo all'inizio dell'esecuzione del Firmware. Questa linea implementa un'impostazione iniziale del Relay.

2° Rung

Non appena la porta P2, facente capo alla luce Verde, è attivata (valore logico 1) si avvia un temporizzatore (T0), inizialmente avente valore logico basso. Dopo 10 secondi (1000 centesimi di secondo) esso assume il valore logico alto.

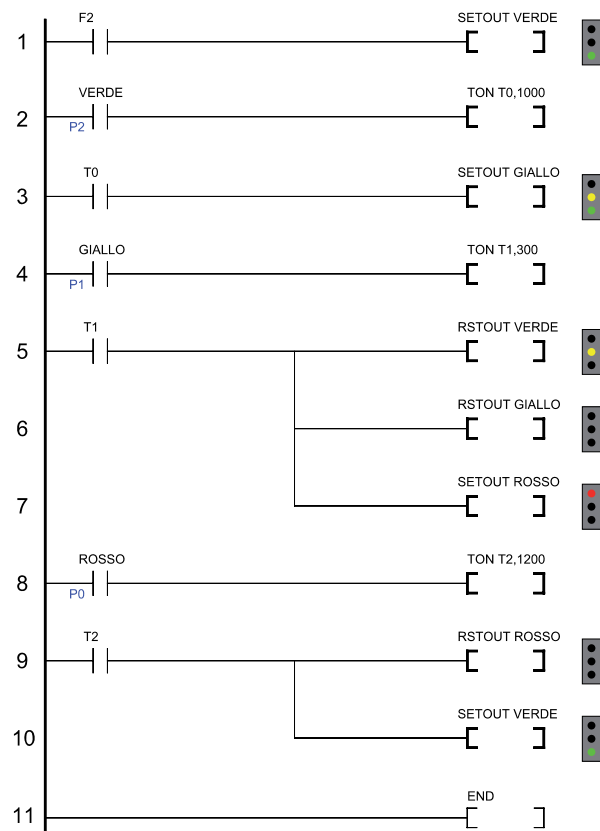



Figura 20
Sorgente Ladder del semaforo

Figura 21
Diagramma di livelli del semaforo

3° Rung

Appena Il timer T0 assume il valore logico alto, segno che sono passati 10 secondi, si attiva, tramite il comando SETOUT, la porta relativa alla luce Gialla.

4° Rung

Non appena la porta P1, facente capo alla luce Gialla, è attiva (valore logico 1) si avvia un altro temporizzatore (T1), inizialmente avente valore logico basso. Dopo 3 secondi (300 centesimi di secondo) esso assume il valore logico alto.

5°, 6° e 7° Rung

Appena Il timer T1 assume il valore logico alto, segno che sono passati 3 secondi, si disattivano, tramite il comando RSTOUT, le porte relative alle luci Verde e Giallo (si noti la classica configurazione in parallelo) e contemporaneamente si attiva, tramite il comando SETOUT, la porta relativa alla luce Rossa.

8° Rung

Non appena la porta P0, facente capo alla luce Rossa, è attiva (valore logico 1) si avvia un ulteriore temporizzatore (T2), inizialmente avente valore logico basso. Dopo 12 secondi (1200 centesimi di secondo) esso assume il valore logico alto.

9° e 10° Rung

Appena Il timer T2 assume il valore logico alto, segno che sono passati 12 secondi, si disattiva, tramite il comando RSTOUT, la porta relativa alla luce Rossa (si noti anche qui la configurazione in parallelo) e contemporaneamente si attiva, tramite il comando SETOUT, la porta relativa alla luce Verde. Il processo in questo modo continua all'infinito, riverificandosi la condizione logica appartenente al Rung 2.

11° Rung

È la linea che chiude il sorgente Ladder.

Naturalmente questa è una delle molteplici soluzioni che possono risolvere un determinato problema. È possibile infatti scrivere un programma in mille modi diversi.

Figura 22
Il tasto per attivare il Time Chart

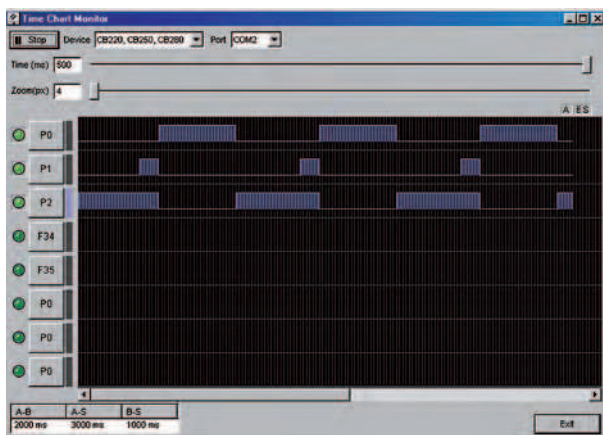



Figura 23
Il Time Chart in funzione

Collaudo

Appena viene fornita l'alimentazione al circuito, i diodi Led si accenderanno rispettando la sequenza e la temporizzazione programmata tramite il linguaggio Ladder. È possibile seguire al video del computer lo stato logico delle porte, attraverso un'altra strepitosa funzione di utilità, il Time Chart. Essa si attiva attraverso il tasto della barra degli strumenti, raffigurato in figura 22. La pressione del pulsante Start contenuto nella videata, avvia l'analisi delle uscite selezionate, proprio come un oscilloscopio o un analizzatore logico (figura 23).

Conclusioni

Si conclude qui questo pratico tutorial sul linguaggio Ladder, ma gli appuntamenti si rinnovano nelle puntate a venire, presentando tanti progetti utili e pratici, utilizzando il Cubloc. Ci sentiamo di dare due piccoli grandi consigli: primo, provate a sperimentare i prototipi proposti negli articoli e ad approfondirli poco per volta; secondo, leggete e rileggete gli articoli tante volte, in modo da fissare bene i concetti e compenetrarvi nella filosofia del linguaggio.

More Info Please!
Inserisci il Codice **267040** alla pagina
www.farelettronica.com/mip
oppure utilizza il **modulo** a pagina **3**

Prima parte: *Luglio/Agosto 2007*
Introduzione al sistema

Seconda parte: *Settembre 2007*
La programmazione in NQC (I)

Terza parte: *Ottobre 2007*
La programmazione in NQC (II)

Quarta parte: *Novembre 2007*
La programmazione in NQC (III)

I sistemi robotici Lego Mindstorms

Nel precedente numero abbiamo introdotto il sistema robotico LEGO MINDSTORMS ed il relativo ambiente di sviluppo ROBOLAB. In questa puntata iniziamo ad esaminare un semplice linguaggio di programmazione che consente di sfruttare appieno la potenzialità di tali sistemi.

Introduzione a NQC

Il sistema robotico LEGO MINDSTORMS permette di costruire una grande varietà di robot, che possono essere programmati per eseguire numerose operazioni. Sfortunatamente, il software ROBOLAB fornito con questo set, benché visualmente accattivante, è piuttosto limitato per quanto riguarda la funzionalità. Per poter usufruire a piena potenza del nostro robot, utilizzeremo un ambiente di programmazione diverso che prende il nome di NQC (Not Quite C), sviluppato da Dave Baum appositamente per la gestione dei robot LEGO.
NQC è un linguaggio di programmazione che prende spunto dal C ANSI, anche se sono state introdotte numerose semplificazioni; è molto più facile programmare un robot LEGO in NQC che un normale computer.
Il linguaggio di programmazione C, descritto da Brian Kernighan e Dennis Ritchie nel testo "The C Programming Language" e successivamente modificato e standardizzato dall'ANSI (American National Standard Institute), è oggi un linguaggio di riferimento per numerosi ambienti di sviluppo.
Il C viene definito come linguaggio di medio livello, in quanto unisce elementi dei linguaggi ad alto livello, quali la gestione dei tipi di dato (la possibilità cioè di operare direttamente su dati complessi come i numeri reali), la portabilità e la modularità (ovvero la scomposizione in parti di programma indipendenti), con funzionalità di quelli a basso livello, quali l'efficienza. Per questo il C può essere considerato un vero linguaggio per programmatori.
NQC è un compilatore free distribuito sotto MPL (Mozilla Public License). È possibile scaricare l'ultima versione di NQC dal sito ufficiale http://bricxcc.sourceforge.net, facendo riferimento all'IDE (Integrated Development Environment - Ambiente Integrato di Sviluppo, cioè software che aiuta i programmatori nello sviluppo) che prende il nome di Bricx Command Center.
L'installazione del Bricx Command Center è abbastanza intuitiva al pari di numerosi altri programmi per Windows e viene effettuata eseguendo il file di setup scaricato. Al termine dell'installazione sarà possibile eseguire l'applicazione selezionando la sua icona, presente nel menu avvio di Windows.
L'interfaccia che si presenterà sarà simile a quella di un normale editor di testi, con menu e pulsanti relativi alla gestione dei files, oltre ad alcuni comandi speciali che consentono di compilare e scaricare i programmi sul robot ed ottenere informazioni da esso.

Premiamo il pulsante New File per far aprire una finestra e scriviamo il seguente programma:

```
task main()
{
  OnFwd(OUT_A);
  OnFwd(OUT_C);
  Wait(400);
  OnRev(OUT_A+OUT_C);
  Wait(400);
  Off(OUT_A+OUT_C);
}
```

In prima approssimazione un programma in NQC può essere schematizzato da una funzione principale di tipo task (compito) chiamata main, che racchiude al suo interno mediante parentesi graffe una sequenza di istruzioni. Ogni istruzione termina con il carattere di punto e virgola e viene generalmente posta su di una riga per migliorare la leggibilità del programma. Il nostro program-

La programmazione in NQC

di Fabio Riscica

ma ha sei istruzioni; analizziamole una alla volta.

OnFwd(OUT_A) ;
Questa istruzione dice al robot di attivare la porta di output A, ovvero il motore connesso alla porta chiamata A sull'RCX, per muoversi avanti. Il motore si muoverà alla velocità massima, a meno che questa non venga preimpostata.

OnFwd(OUT_C) ;
È la stessa istruzione della precedente, ma adesso facciamo partire il motore C. Supponendo che il nostro robot ha i motori collegati a due ruote, a questo punto del programma entrambi i motori stanno girando ed il robot sta avanzando.

Wait(400);
Adesso dobbiamo aspettare un po'. Questa istruzione ci dice di aspettare per quattro secondi. L'argomento, ovvero il numero tra parentesi, fornisce il numero di "ticks" in centesimi di secondo di attesa. Quindi per quattro secondi il programma non farà niente ed il robot continuerà ad avanzare.

OnRev(OUT_A+OUT_C);
Il robot è ora andato abbastanza lontano, quindi gli diciamo di andare nella direzione opposta, cioè all'indietro. Si noti che possiamo settare entrambi i motori in una sola volta usando OUT_A+OUT_C come argomento.

Wait(400);
Aspettiamo altri quattro secondi.

Off(OUT_A+OUT_C);
Ed infine spegniamo entrambi i motori.

Il programma completo muove quindi entrambi i motori in avanti per quattro secondi, quindi indietro per altri quattro secondi, ed infine li spegne.

Una volta scritto il programma, bisogna compilarlo utilizzando l'apposito comando di menù ed inviarlo al robot mediante il collegamento ad infrarossi. Nel caso in cui fossero presenti degli errori, questi verranno segnalati nella parte inferiore della finestra di programmazione.

Cos'è un linguaggio di programmazione

I linguaggi di programmazione sono costituiti da un insieme di parole e di regole, definite in modo formale, che consentono la programmazione di un elaboratore elettronico in modo che questi possa eseguire problemi o più precisamente algoritmi. Si definisce algoritmo una sequenza di istruzioni che risolve un problema in un numero finito di passi.
Il codice sorgente di un programma è costituito da un file di testo, scritto secondo le regole imposte dal linguaggio di programmazione, che per poter essere eseguito dall'elaboratore deve essere tradotto utilizzando un software chiamato compilatore.
E' possibile classificare i linguaggi di programmazione in linguaggi di basso, medio ed alto livello, in base al loro livello di astrazione, cioè a quanto questi siano rivolti alla macchina ovvero all'uomo. Il linguaggio di livello più basso in assoluto prende il nome di linguaggio macchina, è interpretato direttamente dall'elaboratore ed è costituito da sequenze di numeri binari corrispondenti al set di istruzioni del microprocessore. Nel linguaggio macchina si fa riferimento direttamente all'architettura del calcolatore e pertanto si raggiunge il massimo livello di efficienza, sia in termini di velocità di esecuzione che di occupazione di memoria, ma i programmi sono quasi del tutto incomprensibili per il programmatore; gli stessi inoltre non sono portabili, cioè non possono essere eseguiti su di una macchina differente da quella per cui sono stati scritti. I linguaggi di alto livello, d'altra parte, perdono di efficienza, ma consentono un lavoro più agevole per il programmatore, che può implementare l'algoritmo più facilmente. Infatti, le parole chiave di un linguaggio ad alto livello sono espresse generalmente in lingua inglese ed esprimono l'azione svolta.

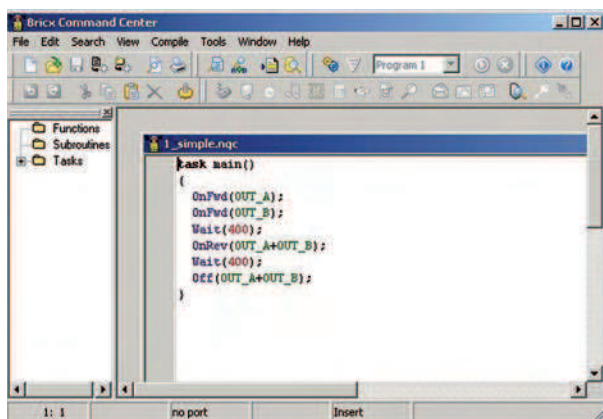



Figura 1
Finestra del Bricx CC

Una volta che il download è completato, agendo sul pulsante Run del modulo RCX il programma passerà in esecuzione ed il robot eseguirà la sequenza di movimenti desiderata.

NQC API: gestione delle uscite

Le uscite possono essere identificate mediante le costanti OUT_A, OUT_B ed OUT_C. Come già accennato, è possibile combinare fra loro le uscite mediante l'operatore di somma.
Si osservi che ogni uscita ha tre differenti attributi: modo, direzione e livello di potenza. Il modo può essere: uscita disattiva (OUT_OFF), uscita attiva (OUT_ON) ed uscita disattiva senza freno (OUT_FLOAT). La direzione può essere: avanti (OUT_FWD), indietro (OUT_REV) ed opposta a quella corrente (OUT_TOGGLE). Il livello di potenza è definito da un numero che può essere compreso tra 0 e 15; è possibile in alternativa utilizzare le costanti OUT_LOW (livello basso), OUT_HALF (livello medio), OUT_FULL (livello massimo).
All'avvio del programma, di default, i tre motori hanno modo OUT_OFF, direzione OUT_FWD e livello OUT_FULL. Nella tabella 1 vengono riportati tutti comandi del linguaggio NQC che possono essere utilizzati per la gestione delle uscite.

Scriviamo un programma più interessante

Vediamo adesso come è possibile far sterzare il nostro robot, arrestando o cambiando la direzione di uno dei due motori. Il programma riportato di seguito fa avanzare il robot e quindi gli fa eseguire una curva a destra.

```
task main()
{
  OnFwd(OUT_A+OUT_C);
  Wait(100);
  OnRev(OUT_C);
  Wait(85);
  Off(OUT_A+OUT_C);
}
```

Si osservi che l'argomento della seconda funzione Wait() influenza l'ampiezza della curva ed è possibile modificare tale valore per eseguire una precisa curva ad angolo retto.
Analogamente all'ANSI C, anche il linguaggio NQC

NQC API – Gestione delle uscite	
SetOutput(uscite,modo)	Imposta le uscite secondo il modo specificato. Esempio: SetOutput(OUT_A+OUT_B,OUT_ON);
SetDirection(uscite,direzione)	Imposta le uscite secondo la direzione specificata. Esempio: SetDirectione(OUT_A,OUT_REV);
SetPower(uscite,livello)	Imposta il livello di potenza delle uscite specificate. Esempio: SetPower(OUT_A,OUT_FULL);
OutputStatus(n)	Restituisce i valori correnti dei settaggi dell'uscita n (n può essere 0,1,2). Esempio: x=OutputStatus(0);
On(uscite)	Attiva le uscite specificate. Esempio: On(OUT_A+OUT_C);
Off(uscite)	Disattiva le uscite specificate. Esempio: Off(OUT_A);
Float(uscite)	Disattiva senza freno le uscite specificate. Esempio: Float(OUT_A);
Fwd(uscite)	Imposta la direzione avanti per le uscite specificate. Esempio: Fwd(OUT_A);
Rev(uscite)	Imposta la direzione indietro per le uscite specificate. Esempio: Rev(OUT_A);
Toggle(uscite)	Inverte la direzione delle uscite specificate. Esempio: Toggle(OUT_A);
OnFwd(uscite)	Attiva ed imposta la direzione avanti per le uscite specificate. Esempio: OnFwd(OUT_A);
OnRev(uscite)	Attiva ed imposta la direzione indietro per le uscite specificate. Esempio: OnRev(OUT_A);
OnFor(uscite,tempo)	Attiva le uscite specificate per il tempo dato. Il tempo è espresso da multipli di 10 mS (100=1 secondo). Esempio: OnFor(OUT_A,75);

Tabella1
I comandi del linguaggio NQC che possono essere utilizzati per la gestione delle uscite

Identificatori, variabili e costanti in ANSI C

Si definisce variabile una locazione di memoria a cui è associato un nome e che è utilizzata per memorizzare un valore che può essere modificato da un programma.
I nomi mediante i quali è possibile fare riferimento alle variabili prendono il nome di identificatori. In ANSI C un identificatore è costituito da uno o più caratteri e può essere di lunghezza arbitraria, anche se sono significativi soltanto i primi 31 caratteri. Il primo carattere deve essere una lettera o un carattere di sottolineatura '_'; può seguire poi un qualsiasi carattere alfanumerico. Si osservi che il compilatore fa distinzione fra lettere minuscole e maiuscole e che un identificatore non può essere uguale ad una parola riservata del linguaggio.
Le costanti sono dei valori che non possono essere alterati durante l'esecuzione del programma. È possibile specificare che una costante è espressa in base esadecimale facendo precedere il valore da '0x'.
Le variabili devono essere dichiarate prima del loro utilizzo nel programma.
La forma di una generica dichiarazione di variabile è

tipo   elenco di variabili ;

dove tipo è uno dei tipi di dato validi del C, eventualmente preceduto da un modificatore, mentre l'elenco di variabili è costituito da uno o più identificatori separati da virgole.
La dichiarazione può avvenire in qualsiasi punto all'interno delle funzioni, anche se in genere conviene adottare la convenzione di dichiarare le variabili prima delle istruzioni.
Le variabili devono essere inizializzate prima del loro utilizzo. Durante la dichiarazione è possibile inizializzare il valore delle variabili utilizzando il simbolo di uguale '=' seguito da una costante.
Le costanti possono essere definite utilizzando la direttiva #define

#define nome valore

dove nome è il nome che identifica nel programma la costante, mentre valore è il relativo valore.

consente l'utilizzo di costanti e variabili secondo una sintassi simile. Le variabili possono però essere soltanto di tipo intero a 16 bit con segno (int). Modifichiamo il nostro ultimo programma utilizzando due costanti, in modo da migliorarne la leggibilità e facilitare la modifica dei valori.

```
#define MOVE_TIME 100
#define TURN_TIME 85

task main()
{
  OnFwd(OUT_A+OUT_C);
  Wait(MOVE_TIME);
  OnRev(OUT_C);
  Wait(TURN_TIME);
  Off(OUT_A+OUT_C);
}
```

In modo equivalente, utilizzando le variabili, si sarebbe potuto scrivere:

```
int move_time;
int turn_time;

task main()
{
  move_time = 100;
  turn_time = 85;
  OnFwd(OUT_A+OUT_C);
  Wait(move_time);
  OnRev(OUT_C);
  Wait(turn_time);
  Off(OUT_A+OUT_C);
}
```

La scelta fatta di rappresentare le costanti in maiuscolo e le variabili in minuscolo è soltanto convenzionale. In genere è comunque preferibile indicare in maiuscolo le costanti, perché queste sono delle vere e proprie macro.

Conclusioni

Abbiamo quindi iniziato la presentazione del linguaggio NQC, appositamente scritto per la programmazione dei robot Lego Mindstorms. Nella prossima puntata procederemo con la trattazione di questo entusiasmante linguaggio di programmazione, approfondendo il concetto di variabile ed illustrando le strutture di controllo che è possibile utilizzare.

LCD dalla A alla Z

In questo articolo svilupperemo le librerie in linguaggio C per l'utilizzo di un display grafico basato su controller T6963C, servendoci del circuito Demograph presentato nella parte IV. Partiremo dalle routine di base (scrittura e lettura di dati e comandi verso il controller) per arrivare a quelle più ad alto livello che ci permetteranno di scrivere e disegnare forme geometriche anche complesse sul display.

Come preannunciato nel precedente articolo, svilupperemo tutto il software per il Demograph in linguaggio C, utilizzando il compilatore MikroC della Mikroelektronica (www.mikroe.com). Il MikroC non è solo un compilatore C, ma è anche un ambiente di sviluppo facile da usare, con editor integrato, colorazione della sintassi, completamento automatico e praticamente tutti gli altri strumenti che caratterizzano un IDE professionale. In Figura 1 è mostrata la finestra dell'ambiente di sviluppo MikroC subito dopo l'apertura del progetto relativo al firmware di collaudo del Demograph.
È possibile scaricare direttamente dal sito del produttore una versione free del MikroC che permette di compilare programmi fino a 2048 istruzioni nel codice eseguibile. Sempre sulla rivista Fare Elettronica è stato pubblicato un tutorial sul MikroC a partire dal n. 252 di Giugno 2006.

Il progetto

Come tutti gli ambienti di sviluppo che si rispettino, il MikroC permette di gestire progetti formati da più file sorgenti e da una serie di impostazioni comuni. Il file principale del progetto ha estensione .ppc e contiene tutte le informazioni utili (i sorgenti che compongono il progetto, il tipo di microcontrollore, la frequenza del clock, i bit della parola di configurazione, ecc). Come progetto di partenza, potete usare quello relativo al firmware di collaudo del Demograph, utilizzato nel precedente articolo per testare il buon funzionamento di tutte le parti del circuito. Il progetto può essere scaricato dalla pagina relativa a questo articolo sul sito di Fare Elettronica. Il file da scaricare si chiama DemographTest.zip e contiene tutti i file necessari per la compilazione. Scompattate questi file in una cartella a vostro piacimento.
È sufficiente scegliere il menu Projects→Open Project e puntare al file DemographTest.ppc, presente nella cartella utilizzata precedentemente, per aprire il progetto.
Potete provare subito a compilare il progetto cliccando sull'icona che rappresenta due ruote dentate nella barra degli strumenti, oppure scegliendo il menu Project→Build o, più semplicemente, premendo Ctrl+F9. Se il processo di compilazione è andato a buon

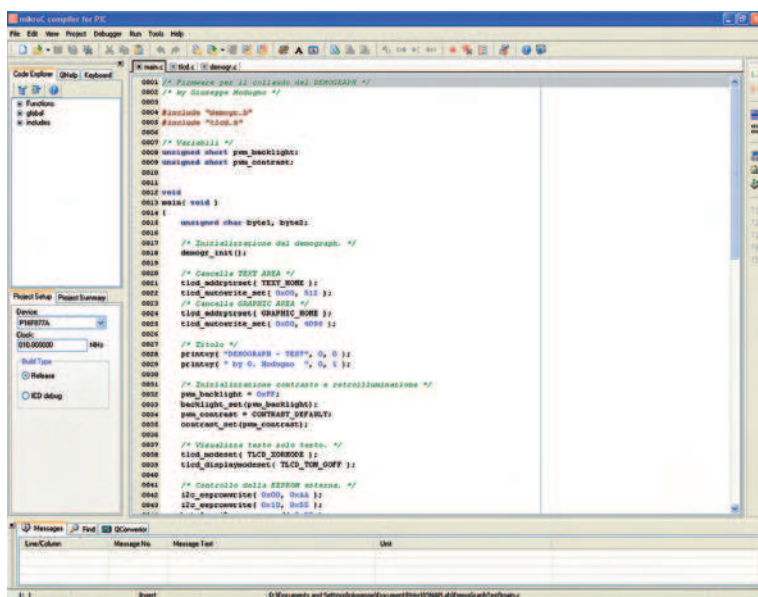



Figura 1
Uno screenshot della finestra del MikroC

La libreria TCLD

di Giuseppe Modugno

fine, nella finestra dei messaggi compare la scritta "Success (Release Build)" in colore verde, con il dettaglio del quantitativo di RAM e ROM utilizzato.

La libreria a basso livello tlcd

In generale, per poter utilizzare un display grafico, dobbiamo "colloquiare" con il suo controller, nel nostro caso il T6963C. Questo "colloquio", di tipo bidirezionale, consiste nell'invio di dati e comandi e nella lettura dello stato del controller. La libreria C che vogliamo sviluppare dovrà, a basso livello, permettere l'invio dei comandi e dei dati e la lettura dello stato. Queste routine sono presenti nei file tlcd.c e tlcd.h. Essendo routine fondamentali per la comunicazione con il display, abbiamo battezzato questa libreria di "basso livello".

Nei successivi paragrafi descriveremo in dettaglio le sezioni più importanti del codice.

I pin di connessione

Com'è noto, per poter modificare lo stato dei pin di una porta, per esempio per mettere a livello basso tutti i pin della PORTD, dovremo utilizzare una istruzione C di questo tipo:

```
PORTD = 0x00;
```

Allo stesso modo, se vogliamo modificare la direzione dei pin di una porta, per esempio configurando tutti i pin come ingressi, dovremo utilizzare quest'altra istruzione:

```
TRISD = 0xFF;
```

Se, anziché modificare lo stato di tutta la porta, vogliamo modificare il livello d'uscita di un singolo pin, per esempio portando alto il pin RD0, l'istruzione sarà:

```
PORTD.F0 = 1;
```

Allo stesso modo, se vogliamo configurare il singolo pin come ingresso:

```
TRISD.F0 = 1;
```

Una libreria di comunicazione con il controller T6963C dovrebbe utilizzare numerose istruzioni di questo tipo, modificando lo stato dei pin di controllo WR, RD e CD e della porta dati. Inoltre, poiché i dati sono bidirezionali, è necessario modificare continuamente la direzione dei pin relativi.

Naturalmente queste istruzioni cambiano a seconda dei pin del microcontrollore utilizzati per la connessione con il controller grafico. Variando queste connessioni, bisognerebbe modificare conseguentemente tutte le istruzioni di questo tipo.

Per evitare ciò e rendere la libreria più generica possibile, all'inizio del file tlcd.h sono definiti i simboli mostrati nel listato 1.

In questo modo, basterà utilizzare `TLCD_DATAPORT`, `TLCD_DATADIR`, `TLCD_WR`,

Listato 1

Prima parte del file tlcd.h.

```
/********** PARTE DA MODIFICARE A SECONDA DELLE CONNESSIONI UTILIZZATE ********/
#define TLCD_DATAPORT           PORTD           /* Porta dati */
#define TLCD_DATADIR            TRISD           /* Registro direzione porta dati */
#define TLCD_WR                 PORTC.F0        /* Pin WR */
#define TLCD_WR_DIR             TRISC.F0        /* Bit di direzione WR */
#define TLCD_RD                 PORTE.F2        /* Pin RD */
#define TLCD_RD_DIR             TRISE.F2        /* Bit di direzione RD */
#define TLCD_CD                 PORTC.F5        /* Pin CD */
#define TLCD_CD_DIR             TRISC.F5        /* Bit di direzione CD */
#define TLCD_XDOTS              128             /* Numero di punti in orizzontale */
#define TLCD_YDOTS              64              /* Numero di punti in verticale */
#define GRAPHIC_HOME            0x0400          /* GRAPHIC HOME Address */
#define TEXT_HOME               0x0000          /* TEXT HOME Address */
#define GRAPHIC_AREA            64              /* GRAPHIC AREA */
#define TEXT_AREA               64              /* TEXT AREA */
//#define WATCHDOG                              /* Commentare se non è usato il watchdog */
/*********************** FINE PARTE DA MODIFICARE ****************************/
```

Listato 2

Seconda parte del file tlcd.h.

```
/* Macro per la gestione della direzione dei PIN (1=Input, 0=Output) */
#define TLCD_DATAPORT_OUT()          TLCD_DATADIR = 0x00
#define TLCD_DATAPORT_IN()           TLCD_DATADIR = 0xFF
#define TLCD_CD_IN()                 TLCD_CD_DIR = 1
#define TLCD_CD_OUT()                TLCD_CD_DIR = 0
#define TLCD_WR_IN()                 TLCD_WR_DIR = 1
#define TLCD_WR_OUT()                TLCD_WR_DIR = 0
#define TLCD_RD_IN()                 TLCD_RD_DIR = 1
#define TLCD_RD_OUT()                TLCD_RD_DIR = 0
```

Listato 3

Funzione `tlcd_busreadstatus()` *[tlcd.c].*

```
unsigned char
tlcd_busreadstatus( void )
{
  /* Ritorna lo stato del display. */
  unsigned char status;

  /* Porta dati in ingresso (lettura). */
  TLCD_DATAPORT_IN();

  /* Setta C_D (comando), resetta_RD (lettura)*/
  TLCD_CD = 1;
  TLCD_RD = 0;

  /* Legge il bus dati e lo memorizza */
  status = TLCD_DATAPORT;

  /* Riporta _RD al normale valore alto. */
  TLCD_RD = 1;

  /* Reimposta il bus dati in output. */
  TLCD_DATAPORT_OUT();

  /* Ritorna lo stato letto. */
  return( status );
}
```

`TLCD_WR_DIR`, ecc. nel codice per renderlo indipendente dalle connessioni utilizzate. Se cambiano i pin, basterà modificare solo questa prima sezione del file tlcd.h.

Naturalmente, i valori inseriti nel file tlcd.h che avete scaricato e che sono visibili nel listato 1 corrispondono alle connessioni usate nel Demograph (si veda lo schema elettrico in Figura 1 della Parte IV). Sempre in questa sezione, è necessario indicare il numero dei pixel orizzontali e verticali del display (`TLCD_XDOTS` e `TLCD_YDOTS`). Nel nostro caso essi sono rispettivamente 128 e 64.

Anche gli indirizzi della VRAM che verranno utilizzati per la TEXT AREA e per la GRAPHIC AREA devono essere specificati in questa sezione di codice (si vedano i simboli `TEXT_HOME` e `GRAPHIC_HOME`). Per il Demograph abbiamo scelto l'indirizzo 0x0000 per la TEXT AREA e l'indirizzo 0x0400 (1024 in decimale) per la GRAPHIC AREA. Oltre all'indirizzo di partenza delle due aree, è necessario definire il numero delle colonne su una riga, sia per la TEXT AREA (simbolo `TEXT_AREA`), sia per la GRAPHIC AREA (simbolo `GRAPHIC_AREA`). Per il Demograph abbiamo scelto un valori pari a 64 per entrambi.

Infine, è possibile definire (togliendo il commento) o meno (lasciando il commento) il simbolo WATCHDOG, a seconda che venga o meno utilizzato il watchdog del microcontrollore. La libreria tlcd deve sapere se è abilitato il watchdog perché deve inserire, durante le varie pause di attesa, le necessarie istruzioni di reset del timer del watchdog, per evitare che il microcontrollore si resetti.

La successiva sezione del file tlcd.h è mostrata nel listato 2.

Qui vengono definite le macro, basate sui simboli visti precedentemente, che verranno realmente utilizzate nel codice e che permettono di configurare i vari pin come ingressi oppure come uscite. Per esempio, per configurare tutti i pin di controllo come uscita e la porta dati come ingresso, basterà scrivere nel codice:

```
TLCD_CD_OUT();
TLCD_WR_OUT();
TLCD_RD_OUT();
TLCD_DATAPORT_IN();
```

Lettura dello stato

Come si è visto nella Parte III, un ruolo chiave nella comunicazione con il controller T6963C è la lettura dello stato. Per esempio, è necessario assicurarsi che i bit STA0 e STA1 del byte di stato siano alti prima di procedere all'invio del successivo comando. La funzione `tlcd_busreadstatus()`, presente nel file tlcd.c e mostrata nel listato 3, legge il byte di stato del controller e ne ritorna il valore.

La funzione è molto semplice e si limita a configurare la porta dati come ingresso e a porre i pin CD e RD rispettivamente alto e basso. Subito dopo è possibile leggere il byte di stato dalla porta dati e memorizzarlo nella variabile `status`. Prima di ritornare, la funzione si preoccupa di riportare i pin dei dati in uscita, così come spiegato successivamente.

In realtà, quello che interessa non è l'intero byte di stato, piuttosto alcuni suoi bit. Nel file tlcd.h si trovano alcune macro che facilitano il controllo delle più comuni condizioni (listato 4).

Listato 4

Macro per il controllo dello stato [tlcd.h].

```
/* Verifica la condizione (STA0==1) &         (STA1==1). */
#define tlcd_checkstatus()                    ((tlcd_busreadstatus()&0x03)==0x03)

/* Verifica la condizione STA2=1. */
#define tlcd_checkstatus2()                   ((tlcd_busreadstatus()&0x04)==0x04)

/* Verifica la condizione STA3=1. */
#define tlcd_checkstatus3()                   ((tlcd_busreadstatus()&0x08)==0x08)

/* Verifica la condizione STA6=1. */
#define tlcd_checkstatus6()                   ((tlcd_busreadstatus()&0x40)==0x40)
```

Listato 5

Funzione tlcd_buswritecommand() [tlcd.c].

```
void
tlcd_buswritecommand( unsigned char cmd )
{
    /* Trasmette un byte comando al display. */

    /* Aspetta l'ACK */
    while( !tlcd_checkstatus() )
#ifdef WATCHDOG
        asm CLRWDT
#endif
        ;

    /* Scrive i dati sul bus */
    TLCD_DATAPORT = cmd;

    /* Resetta le linee di controllo _WR (scrittura) e setta la linea
     * C_D (comando). */
    TLCD_CD = 1;
    TLCD_WR = 0;

    /* Reimposta la linea _WR al valore normale */
    TLCD_WR = 1;
}
```

Listato 6

Funzione tlcd_buswritedata() [tlcd.c].

```
void
tlcd_buswritedata( unsigned char data )
{
    /* Trasmette un byte di dato al display. */

    /* Aspetta l'ACK dal display */
    while( !tlcd_checkstatus() )
#ifdef WATCHDOG
        asm CLRWDT
#endif
        ;

    /* Scrive i dati sul bus */
    TLCD_DATAPORT = data;

    /* Resetta le linee di controllo C_D (dato), _WR (scrittura). */
    TLCD_CD = 0;
    TLCD_WR = 0;

    /* Riporta la linea di controllo _WR alla condizione normale. */
    TLCD_WR = 1;
}
```

Listato 7

Funzione `tlcd_addrptrset()` *[tlcd.c].*

```
void
tlcd_addrptrset( unsigned int addr )
{
     /* ADDRESS POINTER SET.
      * Imposta l'indirizzo nella VRAM da
      * utilizzare nelle successive
      * operazioni
      * di scrittura e lettura.
      */
     tlcd_buswritedata( Lo(addr) );
     tlcd_buswritedata( Hi(addr) );
     tlcd_buswritecommand( 0x24 );
}
```

Per esempio, per aspettare fino a quando i bit STA0 e STA1 siano entrambi ad 1, è sufficiente usare il seguente loop:

```
while( !tlcd_checkstatus() )
     ;
```

Invio di comandi e dati

Come noto, la comunicazione con il controller è caratterizzata dall'invio di comandi e dati da parte del microcontrollore verso il T6963C. Per esempio, per cambiare il puntatore corrente della locazione in VRAM, è necessario inviare prima due byte (byte basso + byte alto) e poi il comando SET ADDRESS POINTER (0x24).
Le routine `tlcd_buswritecommand()` (listato 5) e `tlcd_buswritedata()` (listato 6) si occupano proprio di inviare al controller grafico rispettivamente un comando ed un dato (1 byte).

Listato 8

Funzione tlcd_datawrite() [tlcd.c].

```
void
tlcd_datawrite( unsigned char data, unsigned
                    char incr )
{
     /* DATA WRITE COMMAND.
      * Trasferisce un byte di dato nella
      * memoria del display all'indirizzo
      * VRAM corrente.
      */

     /* Il parametro incr può essere:
      * TLCD_AUTOINCR
      * TLCD_AUTODECR
      * TLCD_AUTOUNCH
      * Questi simboli sono definiti nel
      * file tlcd.h
      */

     tlcd_buswritedata( data );
     tlcd_buswritecommand( 0xC0 | incr );
}
```

Entrambe le funzioni sono molto semplici perché si limitano ad attendere, mediante un ciclo, che il controller sia pronto a ricevere comandi o dati (STA0 e STA1 del byte di stato devono essere entrambi ad 1). Successivamente mettono sulla porta dati il comando oppure il dato da inviare. Infine, impostano il pin WR basso (in entrambi i casi si tratta di una operazione di scrittura) e il pin CD alto (nel caso del comando) o basso (nel caso del dato). Subito dopo il pin WR viene riportato a livello alto per far avvenire effettivamente la scrittura. In realtà, prima di questa istruzione bisognerebbe aspettare almeno 80ns, ma, come già detto nell'articolo precedente, è sufficiente il periodo di elaborazione di una singola istruzione del PIC per rientrare in questo limite temporale. È per questo che non è stata inserita alcuna pausa.
È da notare che nel ciclo di attesa dell'ACK da parte del controller, se è definito il simbolo WATCHDOG, viene azzerato il timer del watchdog (mediante l'istruzione assembler CLRWDT) per evitare che ci sia un reset del PIC.

Scrivere nella VRAM

Sappiamo già che quello che viene visualizzato sull'LCD non è altro che una parte ben precisa della memoria video (la TEXT AREA e/o la GRAPHIC AREA). Per scrivere o disegnare sul display sarà quindi necessario scrivere in determinate locazioni di VRAM. Ancor prima di scrivere, però, sarà necessario modificare il puntatore della VRAM mediante il comando SET ADDRESS POINTER. La routine `tlcd_addrptrset()`, mostrata nel listato 7 e presente nel file tlcd.c, fa proprio questo.
Avendo a disposizione le routine fondamentali `tlcd_buswritedata()` e `tlcd_buswritecommand()`, la routine `tlcd_addrptrset()` è veramente banale: si limita ad inviare prima il byte basso, successivamente il byte alto del nuovo indirizzo VRAM ed infine il comando 0x24, cioè proprio SET ADDRESS POINTER.
Dopo aver impostato l'indirizzo in VRAM, per scrivere un dato in memoria sarà necessario inviare il comando DATA WRITE, cosa che viene fatta dalla routine `tlcd_datawrite()`, mostrata nel listato 8 e presente nel file tlcd.c.
Anche in questo caso, la routine è veramente semplice: si limita ad inviare il byte di dato e successivamente il comando. Quest'ultimo è costruito a partire dal byte `0xC0`, facendo un OR binario con il parametro della funzione (`incr`) che può assumere uno dei tre simboli definiti in tlcd.h (si veda il listato 9).
In pratica, viene inviato il comando `0xC0` nel caso di `TLCD_AUTOINCR` (il puntatore VRAM si incrementa automaticamente), `0xC2` nel caso di `TLCD_AUTODECR` (il puntatore VRAM si decremen-

Listato 9

Definizione dei simboli TLCD_AUTOINCR, TLCD_AUTODECR, TLCD_AUTOUNCH [tlcd.h].

```
/* Modalità d'incremento per le funzioni
    lcd_datawrite() e lcd_dataread(). */
#define TLCD_AUTOINCR     0x00
#define TLCD_AUTODECR     0x02
#define TLCD_AUTOUNCH     0x04
```

ta automaticamente), `0xC4` nel caso di `TLCD_AUTOUNCH` (il puntatore VRAM non viene modificato). Il sistema di definire delle costanti con dei simboli mnemonici è utilizzato ampiamente nella libreria tlcd e permette di richiamare le varie routine in modo più semplice, senza dover ricordare numeri, ma parole o sigle più significative.
Oltre all'operazione di scrittura, è possibile leggere una locazione di memoria video. Essendo però, questa, un'operazione meno comune, ci limitiamo ad indicare il nome della routine associata che è `tlcd_dataread()`.

Autowrite

Un'altra operazione interessante è quella denominata autowrite che permette di scrivere una zona di VRAM, a partire da un indirizzo, inviando solo due comandi: uno all'inizio ed uno alla fine. Questo metodo è molto più veloce rispetto all'invio di tanti comandi di DATA WRITE consecutivi, soprattutto se il numero di byte da scrivere è elevato.
La routine di libreria che implementa l'autowrite è denominata `tlcd_autowrite()`, presente nel file tlcd.c e mostrata nel listato 10.
Questa routine è leggermente più complicata delle altre, ma può essere facilmente capita se ricordiamo la spiegazione del metodo autowrite fatta nella Parte III. La prima operazione è quella di inviare il comando AUTOWRITE (`0xB0`) al controller grafico e, successivamente, aspettare il suo ACK. Subito dopo vengono inviati gli n byte mediante l'istruzione:

```
tlcd_buswritedata( *ptr++ );
```

Naturalmente, prima dell'invio del dato, bisogna aspettare che il controller sia libero, cioè che il suo byte di stato abbia il bit STA3 alto, condizione che viene verificata con la macro `tlcd_checkstatus3()`, dichiarata nel file tlcd.h. Ricordiamo che è necessario resettare il timer del watchodog durante le attese, se questo è utilizzato. Dopo l'invio degli n byte di dati, per terminare l'operazione di autowrite è sufficiente inviare il comando AUTO RESET (`0xB2`).

Listato 10

Routine tlcd_autowrite() [tlcd.c].

```
void
tlcd_autowrite( const unsigned char *ptr, unsigned int n )
{
      /* AUTO WRITE COMMAND.
       * Scrive n byte sul display a partire dall'indirizzo VRAM corrente.
       * I byte da trasmettere si trovano memorizzati a partire dall'indirizzo
       * ptr. Il trasferimento avviene mediante Auto Write, molto più efficiente di
       * tanti comandi DATA WRITE.
       */

      /* Entra nella modalità AUTO WRITE e aspetta l'ack */
      tlcd_buswritecommand( 0xB0 );
      while( !tlcd_checkstatus() )
#ifdef WATCHDOG
              asm CLRWDT
#endif
          ;

      /* Scrive n byte a partire da ptr */
      while( n-->0 ) {
              while( !tlcd_checkstatus3() )
#ifdef WATCHDOG
                      asm CLRWDT
#endif
                      ;
              tlcd_buswritedata( *ptr++ );
      }

      /* Comando AUTO RESET: esce dalla modalità AUTO WRITE */
      tlcd_buswritecommand( 0xB2 );
}
```

Telecamere miniaturizzate

Micro ... solo nelle dimensioni!

Telecamere CMOS con obiettivo nascosto da una vite

	FR419 - € 147,00	FR421 - € 147,00	FR422 - € 147,00	FR308- € 74,00
Tipo	sistema standard PAL (colori)	sistema standard PAL (colori)	sistema standard PAL (colori)	sistema standard PAL (colori)
Elemento sensibile	1/3" CMOS	1/3" CMOS	1/3" CMOS	1/3" CMOS
Risoluzione	380 Linee TV	380 Linee TV	380 Linee TV	380 Linee TV
Sensibilità	3 lux (F1.2)	3 lux (F1.2)	3 lux (F1.2)	1 Lux (F=1.2)
Ottica	f=3,6 mm, F2.0	f=3,6 mm, F2.0	f=3,6 mm, F2.0	f=4,3 mm, F2.0
Alimentazione	12 Vdc	12 Vdc	12 Vdc	5÷12 Vdc
Dimensioni	12 x 12 mm	12 x 12 mm	12 x 12 mm	28 x 24 mm
Forma obiettivo	Vite esagonale con impronta a croce	Vite con impronta a croce a testa bombata	Vite con impronta a croce a testa svasata	-

Fornita in elegante cofanetto. Completa di adattatore video con presa di alimentazione.

Telecamere CMOS a colori

	FR417 - € 120,00	FR418- € 227,00
Tipo	sistema video PAL	sistema video PAL
Elemento sensibile	1/5" CMOS	1/4" CMOS
Risoluzione	240 Linee TV	380 Linee TV
Sensibilità	2 lux (F=1.2)	1,5 lux (F=1.2)
Ottica	f=3,7 mm, F=2.0	f=4 mm, F=2.0
Alimentazione	9÷12 Vdc	2,5÷12 Vdc
Dimensioni	9 x 9 x 20 mm	9 x 9 x 20 mm

Telecamere CMOS con obiettivo nascosto da un bottone

	FR420 - € 147,00	FR309 - € 74,00
Tipo	sistema standard PAL (colori)	sistema standard PAL (colori)
Elemento sensibile	1/3" CMOS	1/3" CMOS
Risoluzione	380 Linee TV	380 Linee TV
Sensibilità	3 lux (F1,2)	1 Lux (con F=1.2)
Ottica	f=3,6 mm, F2.0	f=4,3 mm, F2.0
Alimentazione	12 Vdc	5÷12 Vdc
Dimensioni	12 x 12 mm	28 x 24 mm

Fornita in elegante cofanetto. Completa di adattatore video con presa di alimentazione.

Moduli CMOS a colori

	FR126 - € 52,00	FR126-36 - € 56,00	CAMCOLMHA5 - € 44,00
Tipo	Sistema standard PAL (colori)	Sistema standard PAL (colori)	Sistema standard PAL (colori)
Elemento sensibile	1/3" CMOS	1/3" CMOS	1/3" CMOS
Risoluzione	380 Linee TV	380 Linee TV	380 Linee TV
Sensibilità	4 Lux (F1.4)	3 Lux (F1.2)	1,5 Lux (F2.0)
Ottica	f=5 mm, F4.5 PIN-HOLE	f=3,6 mm, F2.0	f=2,8 mm
Alimentazione	12 Vdc - 20 mA	12 Vdc - 50 mA	8 Vdc - 100 mA
Dimensioni	20,5 x 28 x 17 mm	28 x 17 x 31 mm	26 x 21 x 18 mm

Telecamere CMOS B/N

	FR125 - € 44,00	FR125-36 - € 48,00	FR220 - € 96,00	FR220P - € 125,00	CAMZWCMM1 - € 26,00	FR300 - € 23,00	FR301 - € 27,00
Tipo	sistema standard CCIR (B/N)	sistema standard CCIR (B/N)	sistema standard CCIR (B/N)	sistema standard CCIR (B/N)	sistema standard CCIR (B/N)	sistema standard CCIR (B/N)	sistema standard CCIR (B/N)
Elemento sensibile	1/3" CMOS	1/3" CMOS	1/4" CMOS	1/4" CMOS	1/4" CMOS	1/3" CMOS	1/3" CMOS
Risoluzione	380 Linee TV	380 Linee TV	240 Linee TV	240 Linee TV	380 Linee TV	240 Linee TV	240 Linee TV
Sensibilità	0,2 Lux (F1.2)	0,5 Lux (F1.2)	0,5 Lux (F1.4)	0,5 Lux (F1.4)	0,5 Lux (F1.4)	2 Lux (F1.4)	2 Lux (F1.4)
Ottica	f=4,3 mm, F2.0 PIN-HOLE	f=3,6 mm, F2.0	f=3,5 mm, F2.6 PIN-HOLE	f=3,1 mm, F3.4 PIN-HOLE	f=2,2 mm	f=7,4 mm, F2.8	f=4,9 mm, F2.8
Alimentazione	7 ÷ 12 Vdc - 50 mA	12 Vdc - 50 mA	7÷12 Vdc - 20 mA	7÷12 Vdc - 20 mA	8 Vdc - 100 mA	5 Vdc - 10 mA	5 Vdc - 10 mA
Dimensioni	27,5 x 17 x 18 mm	28 x 17 x 31 mm	8,5 x 8,5 x 15 mm	8,5 x 8,5 x 10 mm	18 x 18 x 17 mm	21 x 21 x 15 mm	16 x 16 x 15 mm

Tutti i prezzi sono da intendersi IVA inclusa

Via Adige, 11 - 21013 Gallarate (VA)
Tel. 0331/799775 - Fax. 0331/778112

Disponibili presso i migliori negozi di elettronica o nel nostro punto vendita di Gallarate (VA).
Caratteristiche tecniche e vendita on-line: *www.futurashop.it*

Codice MIP 267061

I byte da inviare devono essere memorizzati in un array di const unsigned char (il MikroC alloca questo array nella memoria programma, implementandolo come un insieme di RETLW). Alla funzione `tlcd_autowrite()` vengono passati il puntatore a questo array (`ptr`) e la sua dimensione (`n`), cioè il numero di byte da inviare al controller grafico.

Ciò è utile se abbiamo una immagine memorizzata nel PIC che vogliamo ricopiare nella GRAPHIC AREA della VRAM per visualizzarla sul display.
Un'altra operazione tipica che coinvolge l'autowrite è quella di scrivere in una zona della VRAM uno stesso byte, per esempio `0x00`. Questo è utile, per esempio, per azzerare tutta la TEXT AREA e la GRA-

Listato 11

Funzione `tlcd_init()` *presente in tlcd.c.*

```
void
tlcd_init( void )
{
    /* Procedura di inizializzazione del display LCD. */

    /* Imposta la porta dei dati in output. Imposta i pin della porta di
     * controllo in output.
     * N.B.: Le funzioni di questa libreria si aspettano che la porta dati sia
     *       sempre in output. Dopo le operazioni di lettura è necessario
     *       reimpostare la porta dati come output.
     */
    TLCD_DATAPORT_OUT();
    TLCD_CD_OUT();
    TLCD_WR_OUT();
    TLCD_RD_OUT();

    /* Tiene spento il display */
    tlcd_displaymodeset( TLCD_DISPLAY_OFF );

    /* Imposta TEXT AREA, GRAPHIC AREA, TEXT HOME, GRAPHIC HOME */
    tlcd_thaddrset( TEXT_HOME );      /* Imposta TEXT_HOME */
    tlcd_taset( TEXT_AREA );          /* Imposta TEXT_AREA */
    tlcd_ghaddrset( GRAPHIC_HOME );   /* Imposta GRAPHIC_HOME */
    tlcd_gaset( GRAPHIC_AREA );       /* Imposta GRAPHIC_AREA */

    /* Il display deve essere "acceso" (testo e/o grafica) successivamente
     * mediante la lcd_displaymodeset().
     */
}
```

Listato 12

Definizio dei simboli da utilizzare come parametro per la tlcd_displaymodeset() [tlcd.h].

```
/* DISPLAY MODE SET.
 * Imposta la modalità di visualizzazione del display (solo grafica, solo testo,
 * grafica e testo, display off).
 * Il parametro mode può essere
 *   TLCD_TON_GON, TLCD_TON_GOFF, TLCD_TOFF_GON, TLCD_DISPLAY_OFF
 * eventualmente in OR con uno dei seguenti simboli (altrimenti il cursore non
 * viene visualizzato):
 *   TLCD_CURBLINK_ON, TLCD_CURBLINK_OFF
 */
#define TLCD_DISPLAY_OFF        (0x90 | 0x00)
#define TLCD_TON_GOFF           (0x90 | 0x04)
#define TLCD_TOFF_GON           (0x90 | 0x08)
#define TLCD_TON_GON            (0x90 | 0x0C)
#define TLCD_CURBLINK_ON        0x03
#define TLCD_CURBLINK_OFF       0x02
#define tlcd_displaymodeset(mode)    tlcd_buswritecommand(mode)
```

PHIC AREA e quindi cancellare ogni scritta dal display. Se utilizziamo la routine `tlcd_autowrite()`, dovremmo avere un array di byte tutti uguali (per esempio a 0x00), con ovvio spreco di risorse. Per ottimizzare questa situazione, abbiamo creato la routine `tlcd_autowrite_set()`, presente nel file tlcd.c. Per non appesantire la discussione, non inseriamo qui il codice relativo a questa routine, tra l'altro simile a quello della `tlcd_autowrite()`. Ci limitiamo a commentare come deve essere usata, guardando il suo prototipo:

```
void tlcd_autowrite_set( unsigned char data, unsigned int n );
```

Essa è una routine che si aspetta due parametri: il primo è il dato da memorizzare nella VRAM (per esempio, `0x00`), il secondo è il numero di byte da scrivere.

È da notare che, in entrambe le routine di autowrite, deve essere impostato il puntatore VRAM all'indirizzo della prima locazione da scrivere, utilizzando la routine `tlcd_addrptrset()`, prima di richiamare la funzione di autowrite vera e propria.

Inizializzazione del controller

Prima di utilizzare qualsiasi routine della libreria tlcd, è necessario richiamare la funzione `tlcd_init()`, definita nel file tlcd.c e mostrata nel Listato 11.

Per prima cosa, vengono configurati i pin di controllo come uscite e così rimarranno durante tutta la comunicazione con il display. Anche la porta dati viene configurata come uscita, anche se, essendo bidirezionale, la sua direzione potrà cambiare. Come si legge dai commenti, tutte le routine della libreria si aspettano che la porta dati sia già configurata come uscita. Se qualche routine dovesse configurare questa porta come ingresso, prima di uscire dovrà riconfigurarla come uscita.

Subito dopo, viene richiamata la routine `tlcd_displaymodeset()` che invia il comando DISPLAY MODE SET al controller in modo da "spegnere" completamente sia la parte testo che la parte grafica.

Infine vengono richiamate le funzioni `tlcd_thaddrset()`, `tlcd_taset()`, `tlcd_ghaddrset()` e `tlcd_gaset()` per impostare correttamente le sezioni della TEXT AREA e della GRAPHIC AREA della VRAM, così come definite nel file tlcd.h (vedi listato 11).

Nel programma principale bisognerà "accendere" il display, abilitando la parte testo e/o la parte grafica a seconda delle necessità, richiamando nuovamente la funzione `tlcd_displaymodeset()` e

Listato 13

Simboli e macro per l'utilizzo dei pin di I/O [demogr.h].

```
/* Definizione dei PIN del uC usati */
#define PIN_LED1     PORTE.F1     /* LED 1 */
#define PIN_LED2     PORTB.F0     /* LED 2 */
#define PIN_LED3     PORTE.F0     /* LED 3 */
#define PIN_LED4     PORTB.F1     /* LED 4 */

#define PIN_DIP1     PORTA.F5     /* DIP 1 */
#define PIN_DIP2     PORTA.F4     /* DIP 2 */
#define PIN_DIP3     PORTA.F3     /* DIP 3 */
#define PIN_DIP4     PORTA.F2     /* DIP 4 */

#define PIN_KEY1     PORTB.F2     /* KEY 1 */
#define PIN_KEY2     PORTB.F3     /* KEY 2 */
#define PIN_KEY3     PORTB.F4     /* KEY 3 */
#define PIN_KEY4     PORTB.F5     /* KEY 4 */

/* Macro per l'utilizzo dei pin */
#define LED1_ON()    PIN_LED1=1   /* Accendi LED 1 */
#define LED2_ON()    PIN_LED2=1   /* Accendi LED 2 */
#define LED3_ON()    PIN_LED3=1   /* Accendi LED 3 */
#define LED4_ON()    PIN_LED4=1   /* Accendi LED 4 */
#define LED1_OFF()   PIN_LED1=0   /* Spegni LED 1 */
#define LED2_OFF()   PIN_LED2=0   /* Spegni LED 2 */
#define LED3_OFF()   PIN_LED3=0   /* Spegni LED 3 */
#define LED4_OFF()   PIN_LED4=0   /* Spegni LED 4 */

#define DIP1_ON()    (!PIN_DIP1) /* DIP 1 su ON */
#define DIP2_ON()    (!PIN_DIP2) /* DIP 2 su ON */
#define DIP3_ON()    (!PIN_DIP3) /* DIP 3 su ON */
#define DIP4_ON()    (!PIN_DIP4) /* DIP 4 su ON */

#define KEY1()       (!PIN_KEY1) /* KEY 1 premuto */
#define KEY2()       (!PIN_KEY2) /* KEY 2 premuto */
#define KEY3()       (!PIN_KEY3) /* KEY 3 premuto */
#define KEY4()       (!PIN_KEY4) /* KEY 4 premuto */
```

passando come parametro uno dei simboli mostrati nel listato 12.
Per esempio, per "accendere" sia la parte grafica che la parte testo, basterà utilizzare l'istruzione:

```
tlcd_displaymodeset( TLCD_TON_GON );
```

Se si vuole visualizzare anche il cursore del testo, per esempio lampeggiante, bisognerà utilizzare l'istruzione:

```
tlcd_displaymodeset( TLCD_TON_GON | TLCD_CURBLINK_ON );
```

È da notare che questa routine, così come alcune altre, è definita come una macro nel file tlcd.h in quanto coincide semplicemente con l'invio di un opportuno comando (`tlcd_buswritecommand()`).
All'interno dei file tlcd.c e tlcd.h esistono altre funzioni, simboli e macro che non commentiamo in questo articolo per mancanza di spazio e per non appesantire la descrizione. Comunque pensiamo che la spiegazione delle routine fatta precedentemente sia sufficiente per poter capire la restante parte del codice.

La libreria ad alto livello demograph

La libreria tlcd è di basso livello e, come tale, è di fondamentale importanza, ma scomoda da usare. Facendo un paragone con il software di un PC, è come interfacciarsi direttamente ad un driver di una periferica, per esempio di una stampante, piuttosto che sfruttare una API a livello applicazione messa a disposizione dal sistema operativo.
Per questo motivo, abbiamo sviluppato un'altra libreria a più alto livello, dedicata al nostro Demograph. Essa è implementata nei 2 file demograph.c e demograph.h ed utilizzerà la libreria di più basso livello tlcd per il "lavoro sporco" di comunicazione con il controller.

La routine di inizializzazione del Demograph

Analizziamo subito il file include, guardandone un estratto mostrato nel listato 13.
Qui vengono definiti i pin utilizzati per i led, per i dip-switch e per i tasti. Successivamente vengono definite delle macro più comode per accendere/spegnere i led e per leggere lo stato dei tasti e dei dip-switch. Per esempio, volendo accendere il primo led quando è premuto il primo tasto (vi ricordate il firmware di collaudo?), sarà sufficiente la seguente istruzione:

```
if( KEY1() )
    LED1_ON();
else
    LED1_OFF();
```

Passiamo ora alle funzioni presenti nel file demograph.c, in particolare a quella di inizializzazione (`demogr_init()`) mostrata nel listato 14.
Dopo aver disabilitato l'ADC e aver abilitato i pull-up interni sulla PORTB (utili per la lettura dello stato dei tasti), vengono impostati i livelli e le direzioni dei pin del PIC. Subito dopo viene richiamata la routine di inizializzazione del display grafico, già descritta in precedenza, della porta seriale e della porta MSSP (si veda, a tal proposito, la documentazione del MikroC). Più nel dettaglio, la porta seriale viene configurata per una velocità di 9600bps, 8 bit di dati, nessuna parità; la porta MSSP come I2C alla velocità di 100KHz (si ricordi che a questa porta di comunicazione è collegata una EEPROM seriale esterna).
L'ultima operazione è quella di configurazione dei due canali CCP per la generazione dei due segnali PWM utilizzati per la regolazione "software"

della retroilluminazione e del contrasto. Il PIC utilizza internamente il timer TMR2 per la generazione dei segnali PWM, sia per il CCP1 che per il CCP2. Il periodo del segnale PWM (inverso della frequenza) è:

$T_{PWM} = (PR2 + 1) \times 4 \times T_{OSC} \times$ (TMR2 Prescaler)

dove PR2 è il contenuto del registro omonimo, T_{OSC} è l'inverso della frequenza di clock, TMR2 prescaler è il prescaler impostato per il TMR2. Con i valori utilizzati nella funzione `demogr_init()` il periodo è:

$T_{PWM} = (255 + 1) \times 4 \times (1/10000000) \times \times (1) =$ 102,4ms

La frequenza è quindi f_{PWM} = 1/102,4E-6 ~= 10kHz, più che sufficiente per essere filtrata dal passa-basso per il contrasto e per essere invisibile all'occhio per la retroilluminazione. È da sottolineare che, essendo TMR2 usato per entrambi i canali CCP, la frequenza del PWM è la stessa.

I registri CCPRxL configurano, invece, il duty-cycle

Listato 14

Funzione `demogr_init()` *[demogr.c].*

```
void
demogr_init( void )
{
    /* Disabiliazione ADC e abilitazione pull-up interni sulla porta B */
    ADCON1 = 0x06;
    OPTION_REG.NOT_RBPU = 0;

    /* Primo stato dei pin di output (sono commentati solo i pin ad 1) */
    PORTA = 0x00;
    PORTB = 0x00;
    PORTC = 0x01;                /* NOT_WR = 1 */
    PORTD = 0x00;
    PORTE = 0x04;                /* NOT_RD = 1 */

    /* Direzione dei pin di I/O (1=Input - 0=Output) */
    TRISA = 0x3C;                /* RA0, RA1 output. RA2-RA5 input. */
    TRISB = 0xFC;                /* RB0, RB1 output. RB2-RB7 input. */
    TRISC = 0x98;                /* RC0-RC6 output. RC7 input. */
    TRISD = 0x00;                /* RD0-RD7 output. */
    TRISE = 0x00;                /* RE0-RE2 output. */

    /* Inizializzazione del display */
    tlcd_init();

    /* Inizializzazione della USART (asincrona, 8 bit di dati,
     * no parità, 9600bps) */
    Usart_Init( 9600 );

    /* Inizializzazione I2C */
    I2C_Init(100000);
    TRISC = 0x98;                /* Baco nella I2C? */

    /* Il periodo del PWM è dato dalla seguente formula:
     *    T = 4*(PR2+1)*T_osc*TMR2_prescaler
     * Con PR2=255 otteniamo un periodo pari a:
     *    T = 4*256/10M*1 = 102.4us -> F ~= 10kHz
     * Con questo valore è possibile variare il duty-cycle con una variabile
     * ad 8 bit.
     * E' da notare che la frequenza del PWM è uguale sia per CCP1 che per CCP2.
     */
    PR2 = 255;
    /* Retroilluminazione. */
    CCPR2L = 0;           /* Imposta il duty-cycle. */
    CCP2CON = 0x0C;       /* Abilita PWM. */
    /* Contrasto. */
    CCPR1L = 0;           /* Imposta il duty-cycle. */
    CCP1CON = 0x0C;       /* Abilita PWM. */

    /* Abilita il TMR2 associato al PWM */
    T2CON = 0x04;
}
```

Listato 15

Funzioni `backligt_set()` *e* `contrast_set()` *[demogr.c].*

```
void
backlight_set( unsigned short new_backlight )
{
    CCPR2L = new_backlight;
}

void
contrast_set( unsigned short new_contrast )
{
    CCPR1L = new_contrast;
}
```

Listato 16

I simboli per il valore del contrasto [demogr.h].

```
/* Alcuni simboli */
#define CONTRAST_MAX        0x80
#define CONTRAST_MIN        0x40
#define CONTRAST_DEFAULT    0x70
```

che può, quindi, essere impostato differentemente per i due canali. La `demogr_init()` si limita ad impostare un duty-cycle nullo. Per scegliere un duty-cycle diverso sarà sufficiente modificare il valore di questi registri ad 8 bit, da un minimo di 0 ad un massimo di 255. Per esempio, impostando un valore di 100 per il CCPR1L, avremo un duty-cycle di circa 100/255=39% (in realtà, i calcoli sono leggermente più complicati). Volendo, invece, accendere completamente la retroilluminazione, sarà sufficiente assegnare al registro CCPR2L il valore massimo di 255.

Le routine di gestione del contrasto e della retroilluminazione

E allora guardiamo le due routine che servono per impostare il livello del contrasto e della retroilluminazione (listato 15).
Esse sono molto semplici e si limitano proprio ad assegnare il nuovo valore del contrasto o della retroilluminazione al registro CCPR1L o CCPR2L rispettivamente. Il parametro da passare a queste routine sarà un numero compreso tra 0 (valore minimo) e 255 (valore massimo).
Se per la retroilluminazione un qualsiasi valore del parametro può andare bene (da 0 a 255), per il contrasto il discorso è leggermente più complicato. Infatti, ci sono degli intervalli di tensione troppo bassi o troppo alti che determinano un contrasto visivo dell'LCD sicuramente inaccettabile. Per questo sono stati definiti tre simboli nel file include (listato 16) che rappresentano dei valori estremi del parametro `new_contrast` da passare alla routine `contrast_set()` che non dovrebbero essere mai superati.
Inoltre è definito anche un valore ottimale del contrasto che può essere utilizzato come valore di inizializzazione. Tali valori sono stati trovati empiricamente variando il parametro da 0 a 255.

Scrivere testo

Una delle operazioni più comuni che può essere fatta con un display è quella di scrittura di un testo, cioè di una sequenza di lettere. La routine `printxy()`, presente nel file demograph.c e mostrata nel listato 17, implementa proprio la scrittura di una sequenza di caratteri a partire da una posizione (x,y) determinata.
È una routine molto corta (contiene solo 3 istruzioni!), ma può essere di difficile comprensione per alcuni. Ne diamo qui una spiegazione dettagliata.
Come si è già detto, scrivere del testo sul display vuol dire scrivere dei byte nella VRAM, in particolare nella TEXT AREA della VRAM. Il primo byte della TEXT AREA, il cui indirizzo è definito TEXT HOME, rappresenta la codifica del carattere da visualizzare nella posizione (0,0), in alto a sinistra del display. Il secondo byte della TEXT AREA rappresenta la codifica del carattere da visualizzare in posizione (1,0), cioè a destra del primo carattere. E così via per tutti gli altri caratteri della prima riga. Qual è l'indirizzo del primo carattere sulla seconda riga? Sarà quello di indirizzo (TEXT HOME) + (TEXT AREA) (si veda la Parte III per maggiori dettagli). In generale, (TEXT HOME) + (TEXT AREA) × y rappresenta l'indirizzo in VRAM del primo carattere della riga y-esima (la prima riga è associata ad y=0). Se, anziché il primo carattere sulla riga, consideriamo il carattere sulla riga y-esima e sulla colonna x-esima, sarà sufficiente aggiungere all'indirizzo del primo carattere della

Listato 17

Routine `printxy()` *[demogr.c].*

```
void
printxy( const char *str, unsigned short x, unsigned short y )
{
    tlcd_addrptrset( TEXT_HOME+TEXT_AREA*y+x );
    while( *str )
        tlcd_datawrite( *str++-0x20, TLCD_AUTOINCR );
}
```

Listato 18

Firmware di collaudo

```
void
main( void )
{
    unsigned char byte1, byte2;

    /* Inizializzazione del demograph. */
    demogr_init();

    /* Cancella TEXT AREA */
    textarea_clear();
    /* Cancella GRAPHIC AREA */
    grapharea_clear();

    /* Titolo */
    printxy( "DEMOGRAPH - TEST", 0, 0 );
    printxy( " by G. Modugno  ", 0, 1 );

    /* Inizializzazione contrasto e
       retroilluminazione */
    pwm_backlight = 0xFF;
    backlight_set(pwm_backlight);
    pwm_contrast = CONTRAST_DEFAULT;
    contrast_set(pwm_contrast);

    /* Visualizza solo testo. */
    tlcd_modeset( TLCD_TEXTONLY );
    tlcd_displaymodeset( TLCD_TON_GOFF );

    /* Controllo della EEPROM esterna. */
    i2c_eepromwrite( 0x00, 0xAA );
    i2c_eepromwrite( 0x10, 0x55 );
    byte1 = i2c_eepromread( 0x00 );
    byte2 = i2c_eepromread( 0x10 );
    /* Scrive il risultato della EEPROM. */
    if( (byte1==0xAA)&&(byte2==0x55) )
        printxy( "EEPROM: OK", 0, 7 );
    else
        printxy( "EEPROM: ERR", 0, 7 );

    /* CICLO PRINCIPALE */
    while(1) {
        /* Gestione RS232 */
        if( Usart_Data_Ready() )
            Usart_Write( Usart_Read() );

        /* Gestione dei tasti e dei led. */
        if( KEY1() ) {
             LED1_ON();
             pwm_backlight++;
             backlight_set(pwm_backlight);
             Delay_ms(50);
        } else
             LED1_OFF();
...

/* Gestione dip-switch */
if( DIP1_ON() )
    printxy( "DIP1: ON ", 0, 3 );
  else
    printxy( "DIP1: OFF", 0, 3 );
    }
}
```

stessa riga la quantità x. Nell'ottica di quello che abbiamo detto, è semplice capire il significato della prima istruzione della funzione `printxy()`:
`tlcd_addrptrset(TEXT_HOME+TEXT_AREA*y+x)`
Essa si limita a modificare il puntatore corrente della VRAM nell'indirizzo che contiene il carattere in posizione (x,y), posizione dell'LCD da cui dobbiamo iniziare a scrivere il nostro testo. Subito dopo c'è il ciclo che, carattere per carattere, scrive sul display. In C le stringhe di testo terminano con un carattere nullo, quindi la condizione controllata dal while() può essere tradotta come: "esegui le istruzioni nel ciclo se il carattere non è nullo". La variabile `str` viene usata come puntatore che, a partire dal primo carattere, scandisce tutta la stringa fino alla fine. Se la stringa passata come argomento è formata, per esempio, da 4 caratteri (più il carattere nullo), 4 volte verrà eseguita l'istruzione all'interno del ciclo. Questa istruzione è quella che fisicamente scrive in VRAM il byte corrispondente ad un carattere. Come abbiamo avuto modo di notare nella Parte III, la codifica dei caratteri usata dal T6963C è praticamente uguale alla codifica ASCII spostata di 32 posizioni (il carattere ASCII 32 diventa il carattere 0 della codifica del T6963C, il carattere 33 diventa il carattere 1, ecc.). È per questo che alla funzione `tlcd_datawrite()` viene passato, come primo argomento, cioè come byte da scrivere in VRAM, il carattere ASCII (`*str`), decrementato di 0x20 (32 decimale). Il post-incremento (`*str++`) sposta il puntatore della stringa al successivo carattere. Il secondo parametro, `TLCD_AUTOINCR`, indica la modalità di scrittura del dato in VRAM. In questo caso viene indicata la modalità di scrittura e di incremento contestuale del puntatore in VRAM, così da avere già pronto questo puntatore al successivo ciclo.

Un esempio completo

Come semplice esempio di programma completo che utilizza la libreria demograph, quindi la libreria tlcd, prendiamo in esame la funzione principale del firmware di collaudo utilizzato per testare il buon funzionamento del nostro prototipo (Listato 18).
Dopo l'inizializzazione del Demograph, viene cancellata la TEXT e la GRAPHIC AREA in modo che, all'accensione del display (si ricordi che né `tlcd_init()` né `demogr_init()` accendono il display), l'LCD sia "pulito". Le routine di cancellazione delle due aree nella VRAM sono sempre implementate in demograph.c. Successivamente viene scritto un titolo sulle prime due righe del display e viene impostata la massima retroilluminazione e il contrasto di default. Solo dopo viene acceso il display in modalità testo. Il passo successivo è quello di controllare la corretta comunicazione e il buon funzionamento della EEPROM esterna. Il test viene effettuato scrivendo due byte diversi in

due locazioni diverse e leggendo subito dopo tali valori per verificare che tutto sia andato a buon fine. Anche le due routine di lettura/scrittura con la EEPROM esterna (`i2c_eepromread()` e `i2c_eeprom_write()`) sono definite in demograph.c.

Il ciclo indefinito principale si occupa semplicemente di inviare l'echo dei caratteri ricevuti da seriale (`Usart_Data_Ready()`, `Usart_Write()` e `Usart_Read()` sono funzioni di libreria del MikroC) e di leggere lo stato dei tasti e dei dip-switch. Se un tasto viene premuto, si accende il led corrispondente e si incrementa/decrementa il livello della retroilluminazione/contrasto a seconda del tasto premuto. Se il tasto non è premuto, il led corrispondente viene spento. La pausa di 50ms, nel caso di tasto premuto, serve per rallentare l'incremento e il decremento della retroilluminazione e del contrasto per evitare di variare troppo velocemente il livello semplicemente, sfiorando il tasto. Infine, per ogni dip-switch, viene visualizzato sul display il suo stato.

Conclusioni

In questo articolo abbiamo creato, commentando i dettagli fondamentali, la libreria di basso livello tlcd per la comunicazione tra il microcontrollore e il controller grafico T6963C, implementando tutti i comandi supportati. Questa libreria è stata scritta per essere riutilizzata anche con altre schede, oltre al Demograph, dove le interconnesioni tra il display grafico e il PIC16F877A sono diverse. Inoltre, con un po' di impegno, è anche possibile fare un porting della libreria per un altro PIC o addirittura per un microcontrollore di un altro produttore.

Successivamente, abbiamo creato la libreria di più alto livello che, basandosi sulle librerie tlcd, permette di utilizzare facilmente tutte le periferiche del Demograph. Il risultato è stato un codice molto compatto (il 12% della Flash e il 9% della RAM considerando tutto il programma di test del Demograph) e veloce per un microcontrollore a basso costo come il PIC16F877A. Utilizzare un display grafico, quindi, non è molto più complesso che usare display alfanumerici, come si potrebbe ingiustamente immaginare.

Nel prossimo articolo, amplieremo la libreria di alto livello con altre routine grafiche molto interessanti.

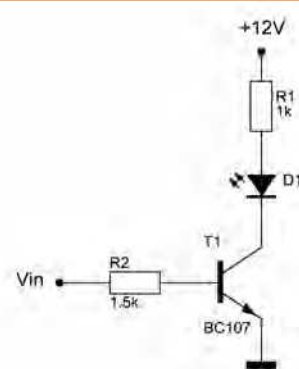
More Info Please!
Inserisci il Codice **267054** alla pagina **www.farelettronica.com/mip** oppure utilizza il **modulo** a pagina **3**

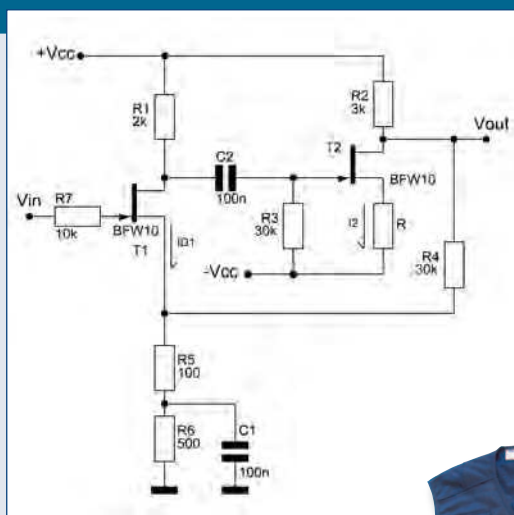
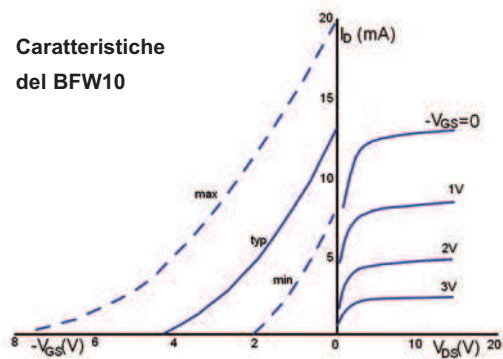
Elettro

RISPONDI & VINCI!

FACILE

La tensione Vin è un'onda quadra 0/5V con periodo 1s e duty cycle del 50%. Detrminare la corrente nel LED in corrispondenza dei due livelli dell'onda in ingresso considerando di 1,2V la caduta su D1.

Se rispondi correttamente potrai vincere il righello in alluminio con calcolatrice a 8 cifre e doppia alimentazione.

DIFFICILE

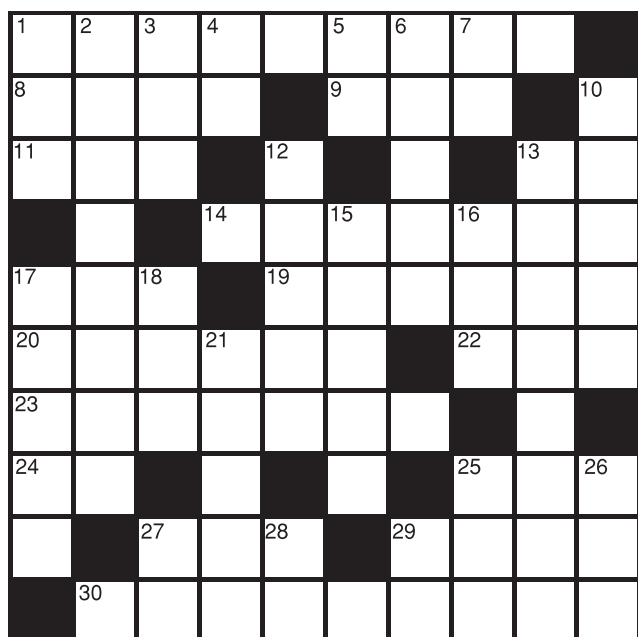
Dato il circuito di figura, determinare il valore di R affinchè si abbia in unscita una tensione nulla in corrispondenza di un ingresso nullo (Vcc=15V).

Per i più bravi in palio il bellissimo gilet reporter di Fare Elettronica.

Le risposte ai quiz "Facile" e "Difficile" vanno inviate esclusivamente compilando il modulo su www.farelettronica.com/eq specificando la parola chiave "Ohm". Le risposte ed i vincitori (previa autorizzazione) sono pubblicati alla pagina www.farelettronica.com/eq a partire dal 15 del mese successivo alla pubblicazione sulla rivista. A tutti i partecipanti verrà assegnato un **buono sconto del 10%** (validità 3 mesi dalla data di assegnazione) utilizzabile per un prossimo acquisto su www.ieshop.it

Quiz

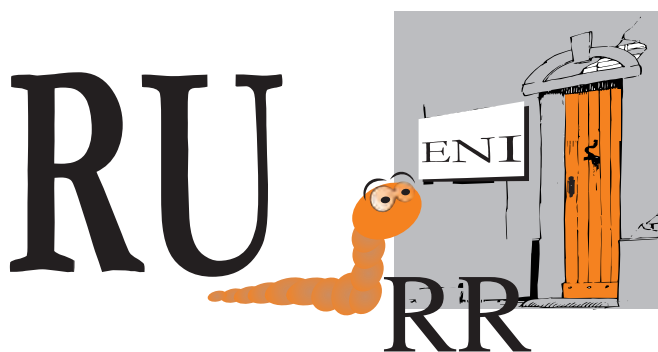
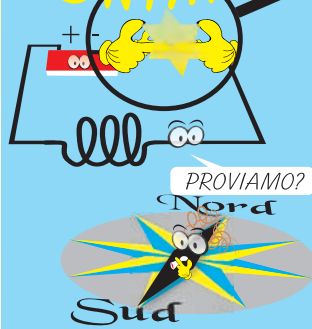
Spazio in collaborazione con www.parole.tv

ELETTRO CRUCIVERBA

1	2	3	4		5	6	7		■
8				■	9			■	10
11			■	12	■		■	13	
■		■	14		15		16		
17		18	■	19					
20			21			■	22		
23							■		■
24		■		■		■	25		26
	■	27		28	■	29			
■	30								

ORIZZONTALI
1. Può essere capacitiva o induttiva
8. Il sorgere del giorno
9. Lo è il metano
11. Precede cognomi scozzesi
13. Il simbolo del ferro
14. Allenatore all'inglese
17. Ambito Territoriale Ottimale (sigla)
19. Nati da incroci, spuri
20. Mezzo byte
22. Vezzo settecentesco delle dame
23. Non legato, agile nei movimenti
24. Il sottoscritto
25. Abiti da frati
27. Lubrificanti
29. Unità di misura della tensione
30. Si pratica prima dell'intervento chirurgico

VERTICALI
1. Un tipo di memoria volatile
2. Flessibile, non rigido
3. Le prime nozioni
4. Taranto (sigla)
5. Agrigento (sigla)
6. È opposto allo zenith
7. Iniziano zona e settore
10. Grave e preoccupante
12. Si accende nel forno per gratinare
13. Gli stati come USA e Brasile
15. L'albero di Natale
16. Anaïs scrittrice di origine francese
17. Preoccupazione
18. La cintura del kimono
21. Matematico pioniere dell'elettronica digitale
25. Richiesta di aiuto
26. La prima metà d'Italia
27. Il contrario di off
28. Isernia (sigla)
29. Venezia (sigla)

ELETTRO REBUS

FRASE: (12)

ELETTROON

Progettare un sistema embedded

Le architetture basate su standard aperti stanno dominando il mercato dei Computer-On-Module. ETX, XTX e COM Express, secondo stime attendibili, deterranno il 70% del mercato dei moduli COM entro la fine di quest'anno. Ecco di cosa si tratta e come queste tecnologie possono dare flessibilità e potenza allo sviluppo di sistemi embedded.

Cosa si intende per Computer-On-Module?

Single-board computer

Un microcomputer che incorpora, in un unico blocco, funzionalità che nei sistemi tradizionali sono distribuite su blocchi fisicamente separati è detto embedded. Spesso il termine indica l'intera applicazione che nasconde al suo interno il microcomputer. Una forma diffusa di microcomputer embedded è il Single Board Computer (SBC), che incorpora CPU, controllers di I/O e tutti i connettori standard su un'unica piccola piastra (le dimensioni, espresse in mm, di alcuni di essi sono riportate in Figura 1). Quando invece tutte le funzioni sono implementate su singolo chip, questo è detto System On Chip (SOC).

Un *Computer-On-Module* (spesso abbreviato con la sigla COM) è un tipo particolare di SBC. Il modulo COM è un computer completo costruito su di un'unica scheda. Il progetto è focalizzato su un singolo microprocessore con RAM, pin di input/output e tutte le altre periferiche tipiche di un PC. Diversamente da un SBC, però, un COM non dispone fisicamente dei connettori standard di un PC (come seriale, USB, parallela, PCI) direttamente sulla scheda. Invece, le linee di tali periferiche sono portate su file di connettori standard. Il modulo potrà quindi essere montato su qualunque scheda madre che rispetti lo standard di quel COM. Il principale vantaggio che ne deriva è la possibilità di realizzare un sistema integrato dalla dimensioni contenute, con un time-to-market ridotto, ma con la potenza di un vero personal computer.

Esistono vari tipi di standard COM, tra questi si farà riferimento ad alcuni dei più promettenti:

- ETX (Embedded Technology eXtended);
- XTX (eXpress Technology for ETX);
- COMExpress (Computer-On-Module Express).

In sostanza, tali standard identificano il fattore di forma ed il pinout della scheda.

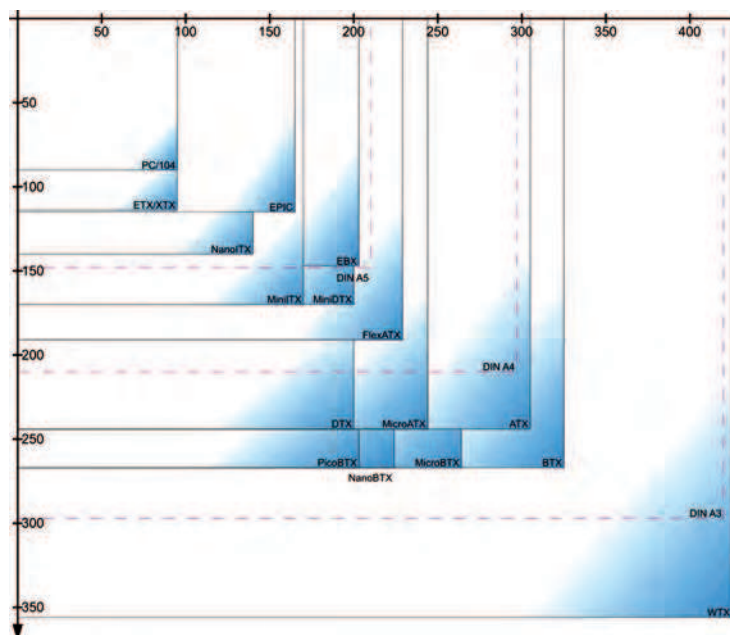



Figura 1
Il processo di standardizzazione ha interessato non solo il progetto elettronico, ma anche e soprattutto la forma, le dimensioni e il posizionamento dei componenti: nel tempo si sono affermati alcuni formati standard a cui si sono conformati tutti i produttori

con i moduli COM

di Savino Giusto

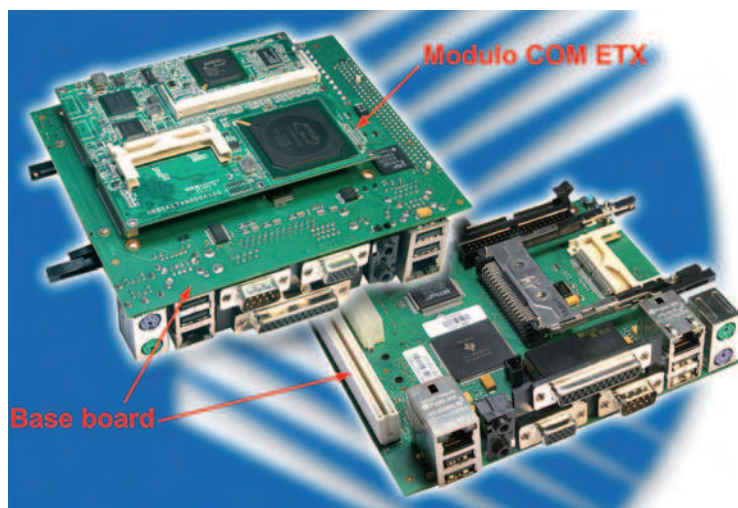



Figura 2
Un Computer-On-Module è un tipo particolare di Single-Board-Computer. Si tratta in sostanza di un sistema embedded con la stessa potenza di calcolo di un personal computer, ma dalla dimensioni ridottissime. ETX rappresenta un particolare fattore di forma dei moduli COM

In Figura 2 è mostrato un esempio di modulo COM in standard ETX montato su di una scheda madre, detta base board. Le dimensioni ridottissime (poche decine di centimetri) consento di realizzare un sistema embedded in uno spazio davvero ridotto e personalizzare l'applicazione sulla base delle necessità di progetto. Soluzioni di questo tipo sono utilizzate in tutte quelle applicazioni in cui la potenza di calcolo offerta da un semplice sistema a microcontrollore non risulta sufficiente. Si ricorre quindi alla potenza e flessibilità di tali sistemi su cui "gira" un vero e proprio sistema operativo, come Window CE, Windows Xp Embedded o varie distribuzioni Linux.

La necessità di uno standard

Con la continua evoluzione dei processori e di bus sempre più veloci, gli sviluppatori di sistemi embedded si devono scontrare con il problema di come implementare queste nuove tecnologie all'interno dei loro progetti, garantendo al contempo un valore aggiunto che permetta ai loro prodotti di guadagnare quote di mercato.

Sviluppare ed ingegnerizzare su una nuova scheda, nuovi processori e bus di comunicazione diventa così estremamente costoso. In tal senso, la metodologia dei Computer-On-Module è diventata una tecnica ben accettata per consentire ai progettisti di superare questi ostacoli. I COM possono essere semplicemente definiti come moduli che contengono tutti i componenti necessari per un host computer, ingegnerizzato in un componente cosiddetto off-the-shelf (spesso abbreviato COTS, acronimo di Components Off-The-Shelf); il termine si riferisce a componenti hardware (ma anche software) disponibili sul mercato per l'acquisto da parte di aziende di sviluppo interessate a utilizzarli nei loro progetti. L'uso di componenti COTS hardware rappresenta una possibilità consolidata. Il modulo COM è sempre lo stesso, mentre ciò che cambia è la carrier board, ossia la scheda madre che su cui è inserito, come illustrato nella Figura 2. Insieme, COM e carrier board, garantiscono le funzionalità di un SBC.

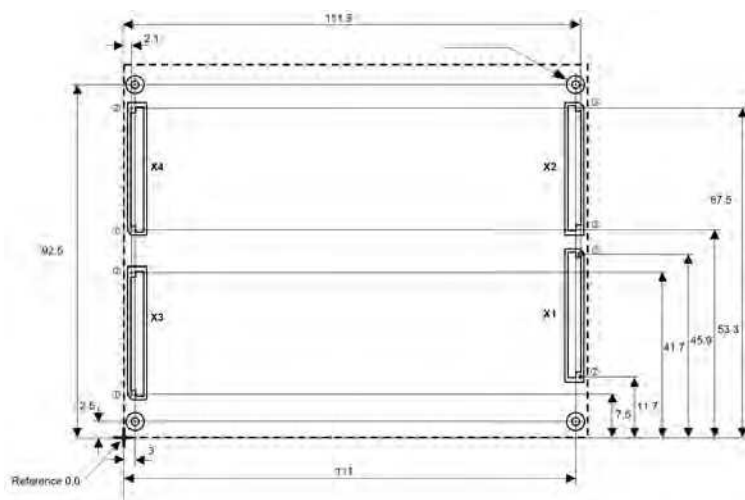



Figura 3
Lo standard ETX prevede una scheda di dimensione 95 × 114 mm, dotata di quattro connettori X1, X2, X3 e X4. Tali connettori servono per rendere disponibile all'esterno le periferiche del modulo COM

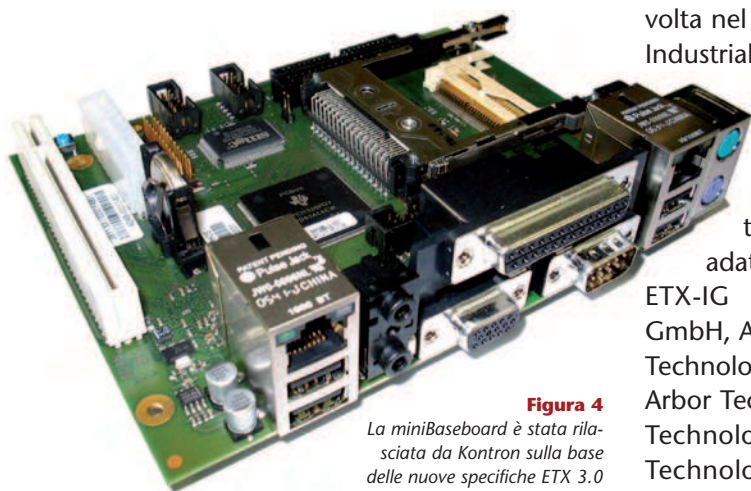



Figura 4
La miniBaseboard è stata rilasciata da Kontron sulla base delle nuove specifiche ETX 3.0

Sono disponibili tutte le funzioni di un PC come grafica, Ethernet, suono, IDE, FDD (Floppy Disk Device), tastiera e mouse, porta parallela, seriale, USB e bus ISA, PCI e PCIe (PCI Express).
Il risultato è un sistema compatto, altamente prestante, con un time-to-market estremamente ridotto. Quando i COM hanno cominciato a guadagnare popolarità, l'industria embedded ha riconosciuto la necessità di uno standard, con lo scopo di garantire modularità e sistemi off-the-shelf. Alcuni dei vantaggi della standardizzazione sono:

- Riduzione dei costi;
- Miglioramento della qualità;
- Architettura standard;
- Scalabilità e flessibilità.

Gli standard descritti in quest'articolo sono tra quelli che hanno riscosso il successo maggiore tra gli sviluppatori di sistemi embedded. In particolare, lo standard ETX è stato l'archetipo di quelli che si sono poi avvicendati, come XTX e COMExpress.

Lo standard ETX

Lo standard ETX [1], acronimo di Embedded Technology eXtended, fu introdotto per la prima volta nel gennaio del 2000 sotto la guida dell'ETX Industrial Group (ETX-IG) [3] per fornire uno standard aperto per applicazioni industriali. Si tratta di un'associazione di compagnie che lavorano per scambiare informazioni riguardo ad ETX, monitorare le tendenze e assicurare che lo standard si adatti pienamente alla esigenze di mercato. ETX-IG include Kontron Embedded Modules GmbH, Advantech, MSC Vertriebs GmbH, ADLINK Technology, Evalue Technology Inc., SECO Srl, Arbor Technology Corp., Axiomtek Co., Blue Chip Technology, AAEON Technology Inc., AEWIN Technologies Co., Honeywell – CMSS and American Portwell Technology.
Lo standard è rivolto ad applicazioni nel campo medicale, gaming & entertainment, militare e aerospaziale.

Il modulo ETX è tra i più compatti (appena 95mm × 114mm) ed integrati. Tutti i moduli che rispettino lo standard sono dotati di 4 connettori che consentono di portare all'esterno i tipici segnali di un personal computer:

1. Connettore X1: PCI bus, USB Audio;
2. Connettore X2: ISA BUS;
3. Connettore X3: VGA, LCD, Video, COM 1, COM2, LPT/Floppy, IrDA, Mouse, Keyboard;
4. Connettore X4: IDE 1, IDE 2, Ethernet ed altri.

La Figura 3 fornisce le caratteristiche geometriche della scheda, mentre in Tabella 1 sono riportare le principali caratteristiche confrontate con gli standard XTX e COM Express.

È ovvio che con l'evolvere della tecnologia, tali moduli devono essere in grado di svilupparsi proporzionalmente, adattandosi alla nuove esigenze di mercato. Il successo dello standard ETX è anche comprovato dal fatto che nell'aprile 2006, l'ETX-IG ha rilasciato l'ultima generazione delle specifiche

CONFRONTO TRA GLI STANDARD PER MODULI COM		
ETX	XTX	COM Express
95 × 114 mm	95 × 114 mm	110 ×155 (extended) 95 × 125 mm (basic) 95 x 95 (compact)
Full legacy (1)	Legacy escluso ISA	Legacy escluso ISA
PCI, ISA	PCI, LPC, PCI Express	LPC, PCI Express
10/100 Ethernet	10/100 Ethernet	10/100, GbE
(2) IDE	(2) IDE, SATA	SATA
(4) USB	(6) USB	(8) USB
AC'97	AC'97 + High Def	AC'97 + High Def

Tabella 1
Confronto tra lo standard ETX, XTX e COM Express. (1) Con il termine "Legacy" sono definite le vecchie periferiche per PC, come ISA bus, RS-232 serial, IEEE 1284 parallel, floppy, e PS/2 keyboard/mouse

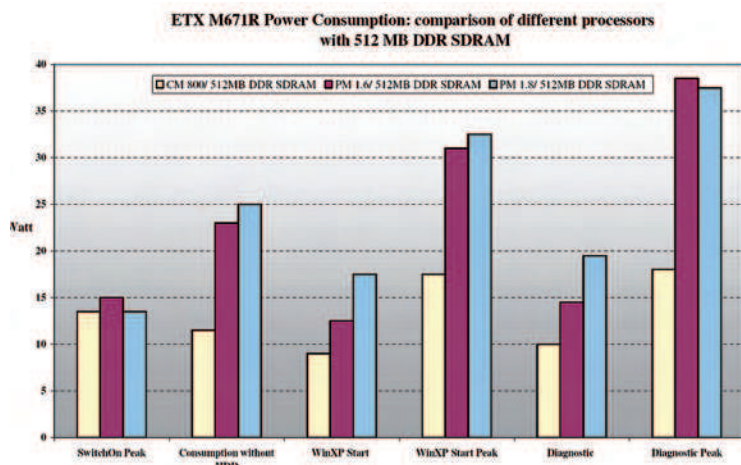



Figura 5
I consumi di potenza dei moduli ETX sono davvero ridotti. Il grafico presenta un confronto fra tre differenti tipi di processori ciascuno dotato di 512MB di DDR SDRAM. I consumi sono differenziati sulla base della modalità operativa (Fonte http://www.seco.it/)

ETX [4], giunte alla versione 3.0.
Il nuovo standard offre tutti i benefici di quello originale con l'aggiunta dell'interfaccia Serial ATA 2x. Tale periferica non modifica assolutamente il pinout preesistente, rendendo il nuovo modulo perfettamente compatibile con le precedenti versioni. Quindi ETX 3.0 non costringe i progettisti a cambiare la base board utilizzata. L'altra novità introdotta è l'aggiunta della USB 2.0 sui preesistenti connettori ETX.

Recentemente rilasciata da Kontron, la miniBaseboard rappresenta un'ottima scheda di sviluppo per applicazioni basate sullo standard ETX 3.0 (Figura 4).

Trattandosi di moduli integrati, particolare attenzione deve essere prestata, in fase di progettazione, ai consumi di potenza (Figura 5).

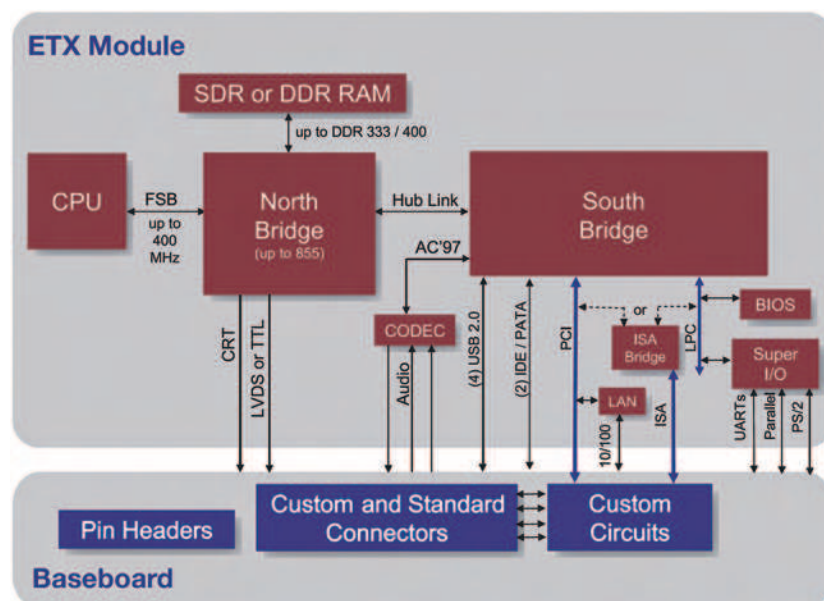



Figura 6
Schema a blocchi di una scheda ETX

La Figura 6 mostra lo schema a blocchi di un scheda ETX, confrontato con quello di un tipico SBC in Figura 7. In particolare si nota, oltre a CPU e RAM, la presenza di due chipset denominati NorthBridge e SouthBridge. Il northbridge (dall'inglese ponte nord), conosciuto anche come Memory Controller Hub (MCH), costituisce, insieme al Southbridge, la parte logica (chipset) di una scheda madre. Il chipset montato sulla scheda madre, viene quindi separato in due parti: Southbridge e Northbridge; sebbene ci siano delle istanze che collegano i due bridge è comodo dividere in due parti il chipset per ridurre al minimo effetti di collo di bottiglia. In Figura 6 sono indicati in rosso tutti i componenti presenti sul modulo COM, mentre in blu quelli che sono posizionati sulla carrier board. Come si nota, invece, in Figura 7 tutti i blocchi logici sono di colore rosso, poiché si tratta di un SBC, ossia un'unica scheda su cui sono montati tutti i componenti (connettori compresi).

Evoluzioni di ETX: lo standard XTX

Congatec ha sviluppato il nuovo standard XTX, evoluzione naturale dell'affermata tecnologia ETX, che elimina il bus ISA per fare spazio a bus seriali come PCI Express e Serial ATA, senza modificare la meccanica della scheda. Per le specifiche complete si consulti l'indirizzo [9].

Il suo principale vantaggio è quello di rendere disponibile le moderne tecnologie di I/O in un fattore di forma già ben collaudato. Mentre con lo standard ETX il connettore X2 è riservato al bus ISA, con XTX tale slot è riservato a bus ad alta velocità come PCI Express (che garantisce una banda di 2.5GHz) e Serial ATA. Gli altri connettori X1, X3 e X4 restano invariati e quindi ancora conformi alle specifiche ETX.

Un'altra richiesta messa a punto con XTX è lo sviluppo di un BIOS capace di supportare tutti gli standard industriali, progettando anche l'architettura di controllo adatta a gestirlo.

Inoltre, sono stati risolti tutti i problemi relativi alla sicurezza dei dati: lo standard XTX supporta

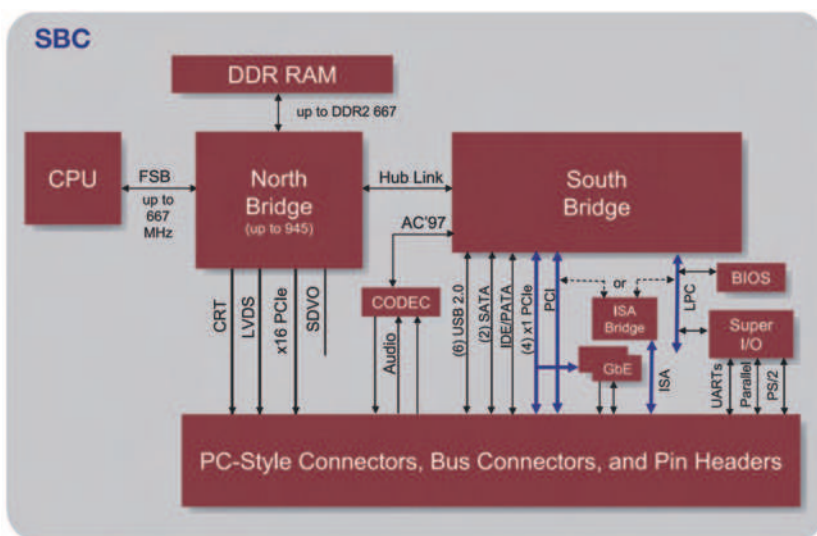



Figura 7
Schema a blocchi di un SBC

Figura 8
Logo dello standard XTX

infatti le specifiche TPM (acronimo di Trusted Platform Module). Il TPM (letteralmente significa modulo per una piattaforma fidata) è un chip le cui specifiche sono pubblicate dal Trusted Computing Group [6]. Esso era noto in precedenza come Chip Fritz, nome che deriva dal cognome del senatore statunitense Fritz Hollings forte sostenitore di questo progetto. Lo scopo principale del TPM è facilitare la memorizzazione sicura dei dati. Il TPM viene generalmente saldato sulla scheda madre di un PC o di un computer portatile dai produttori, ma si può trovare anche nei PDA e in altri dispositivi. Molti dei portatili prodotti nel 2006 sono dotati di TPM. Secondo la International Data Corporation, entro il 2010 tutti i portatili e praticamente quasi tutti i PC saranno dotati di TPM. Alcune delle sue funzioni sono memorizzazione di chiavi, certificati digitali, password e dati sensibili; protezione da virus, worms ed altri attacchi provenienti dalla rete; protezione dell'identità digitale e firme digitali.

In Figura 10 è mostrato lo schema a blocchi di un tipico modulo XTX. Le scritte in rosso si riferiscono a funzionalità aggiuntive rispetto allo standard ETX.

A partire dal 2006 si è assistito all'affermazione delle soluzioni dual core in grado di soddisfare una richiesta di moduli dalle prestazioni particolarmente spinte che fino ad allora era rimasta inevasa. Un esempio è rappresentato dal modello X945/64 presentato da Congatec [7]. Esso è equipaggiato con un processore Core 2 Duo a 64bit di Intel a 1.5GHz e dotato di 4Mbyte di cache di secondo livello (L2), con possibilità di indirizzare fino a 2Gbyte di RAM DDR2. Un'immagine del modulo è riportata in Figura 11.

Evoluzioni di ETX: lo standard COMExpress

COMExpress è un altro esempio di evoluzione dello standard ETX. Si tratta di uno standard industriale aperto, pensato appositamente per il mondo COM e standardizzato da PICMG [8] (Pci Industrial Computer Manufacturers Group).
Le specifiche prevedono tre formati:

- Compact (95 × 95mm), particolarmente indicato per CPU di tipo LV (Low Voltage) e ULV (Ultra Low Voltage) e chipset di dimensioni limitate.
- Basic (125 × 95mm), per chipset e CPU più prestanti.
- Extended (155 × 95mm), particolarmente adatto per applicazioni dual processor.

COMExpress prevede cinque differenti tipologie di connettori (per le specifiche complete si consulti l'indirizzo [10]) e quindi a differenza di XTX non risulta compatibile con lo standard ETX. Tra questi il più diffuso è il tipo 2, caratterizzato da un'interfaccia grafica PCIe x16. Applicazioni tipiche sono gaming e elaborazione grafica e prevedono l'interfacciamento con schede grafiche ATI e NVIDIA. Per

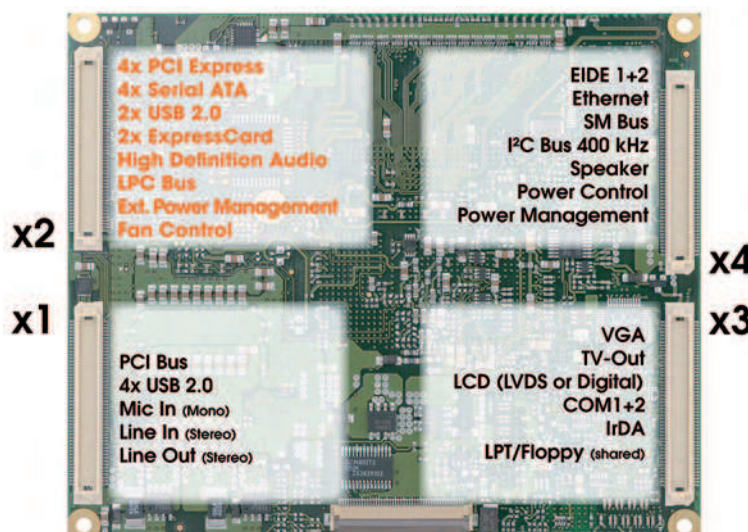



Figura 9
Lo standard XTX mantiene le stesse dimensioni meccaniche di ETX, ma sostituisce il bus ISA sul connettore X2 con PCIe e Serial ATA. In arancio sono evidenziate tali differenze

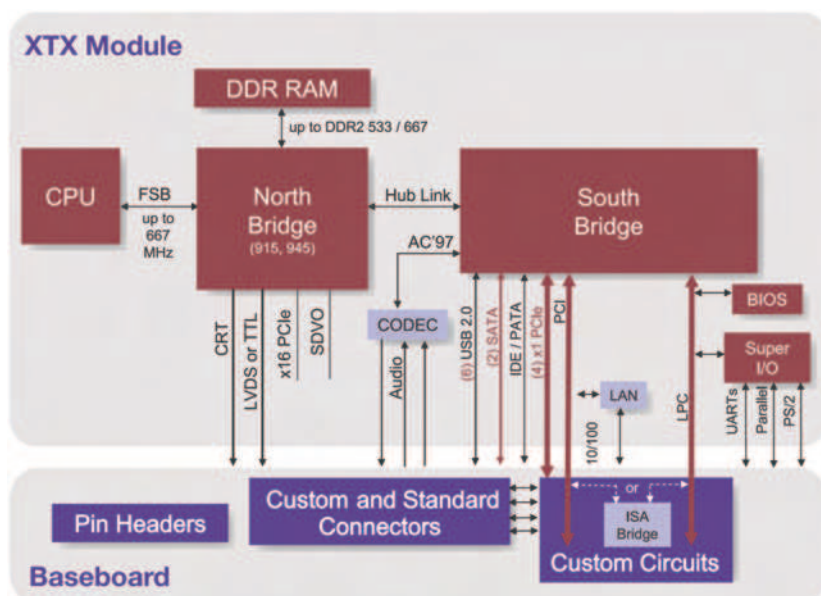



Figura 10
Schema a blocchi dell'elettronica a bordo di una scheda XTX. In rosso sono riportati i componenti presenti sul modulo COM"

gli altri collegamenti tra modulo e scheda madre si faccia riferimento alla Figura 12. Per informazioni sul nuovo standard PCI Express consultare l'approfondimento "Novità del PCI Express".
Da notare anche la tendenza di COMExpress di eliminare interfacce obsolete come la UART. In realtà, tale periferica è ancora molto utilizzata in settori industriale, medicale e militare. Per questi sistemi è preferibile ricorrere allora ad XTX. Un'altra differenza è nella tensione di alimentazione: 5V per ETX/XTX e 12V per COMExpress.

In definitiva, migrare verso una soluzione COMExpress partendo da ETX significa ridisegnare completamente la propria base board. In realtà i vantaggi di tale soluzione saranno più evidenti tra qualche anno quando saranno scomparsi completamente i bus IDE e PCI a vantaggio di PCI Express e Serial ATA.

Figura 11
Il modulo X945/64 è uno dei primi esempi di soluzioni dual core presentato da Congatec

Le piattaforme software dei sistemi embedded

Quando si lavora con sistemi embedded è necessario utilizzare sistemi operativi real-time.
Un esempio è rappresentato dalla nuova versione di Windows Embedded CE 6.0 [11]. Si tratta di una variante del classico sistema operativo Microsoft, indicato per tutti i sistemi embedded. Esso supporta processori Intel x86 e compatibili, MIPS, ARM e Hitachi SuperH. Windows CE (una tipica schermata è mostrata in Figura 13) è ottimizzato per dispositivi con un minimo quantitativo di memoria, infatti necessita di meno di 1Mbyte di memoria. L'occupazione di risorse di un kernel real-time è spesso indicata con il termine tecnico di footprint. E' conforme alle definizione di un RTOS (Real Time Operating System) con una latenza di interrupt deterministica. Supporta fino a 256 livelli di priorità e l'unità fondamentale di esecuzione è chiamata thread. Questo consente al sistema di semplificare l'interfaccia e migliorare il tempo di esecuzione.

Windows CE non è affatto l'unica soluzione per sistemi embedded. Tra le altre si possono ricordare VxWorks, iTRON, eCos, OSE, QNX, Symbian OS, Micro-C/OS II e le distribuzioni di Linux embedded µClinux [12].
In particolare, l'obiettivo che si è cercato di ottenere con µClinux è quello di sopperire alla mancanza, nel settore embedded, di una distribuzione Linux. Questo sistema operativo è sottoposto ai vincoli della licenza GPL che consente la modifica del codice sorgente: nessun vincolo può essere imposto dal distributore, come nessun tipo di licenza a pagamento può essere richiesta. Questo tipo di distribuzione è col tempo diventata l'arma vincente di questa soluzione, poiché ha permesso di adattare completamente il sistema operativo in modo che risponda perfettamente alle singole necessità. Per chiunque volesse approfondire l'argomento è possibile far riferimento all'articolo di Firmware n.16 "µClinux una soluzione per sistemi embedded".

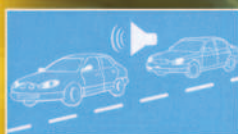
Conclusioni

Questi ultimi anni sono stati caratterizzati dall'affermazione dei cosiddetti moduli COM ed in particolare degli standard ETX, XTX e COM Express. Si tratta di schede dalle ridotte dimensioni e particolarmente adatte per la realizzazione

Più sicurezza in auto con Road Matrix™

Altissima tecnologia al servizio della sicurezza dell'automobilista. Dotato di GPS, telecamera e registratore video, segnala quando si oltrepassa la linea bianca che delimita la carreggiata, registra le immagini in caso di incidente e, in coda, avvisa quando il veicolo che precede riprende la marcia.

Allarme superamento striscia bianca

Registrazione video in caso d'incidente

Segnalazione avanzamento del veicolo che precede

499.00
IVA inclusa

Road Matrix™, tecnologia all'avanguardia in auto

Questo sistema utilizza una telecamera il cui scopo è quello di monitorare continuamente la strada e le linee di demarcazione della corsia impegnata attivando un allarme acustico in caso di colpi di sonno o pericolose disattenzioni. Le immagini riprese dalla telecamera vengono registrate su memoria digitale in modo da offrire, in caso di incidente, una sequenza video dello stesso (12 secondi prima e 6 secondi dopo, attivazione automatica mediante sensore d'impatto). Le immagini riprese vengono anche utilizzate per avvisare, in caso di coda, quando il veicolo che precede riprende la marcia. Tutte queste funzioni sono assistite da GPS, che rileva la velocità dell'autovettura, e da speech processor che genera gli avvisi vocali.

Segnalazione LDWS (Line Departure Warning System)

Quando il veicolo si sposta troppo su un lato della corsia (con il rischio di oltrepassare la linea bianca di demarcazione), il dispositivo genera un allarme sonoro; tale funzione si attiva automaticamente quando si supera una specifica velocità (rilevata dal GPS).

Black Box - Registrazione video

Le immagini riprese dalla telecamera vengono registrate in loop su una memoria digitale. In caso d'incidente un sensore inerziale blocca la registrazione in modo da mantenere in memoria una sequenza di 18 secondi (12 secondi prima e 6 secondi dopo l'impatto). Oltre alle immagini vengono registrati i dati forniti dal GPS (ora, velocità, ecc.)

Avviso avanzamento colonna

In caso di coda per rallentamenti o di fermata ad un semaforo, il sistema avvisa vocalmente quando il veicolo che precede riprende la marcia se entro due secondi anche la vostra vettura non si rimette in moto.

Avviso di guida prolungata e altre funzioni

Ogni ora il sistema informa vocalmente da quanto tempo state guidando in modo da poter valutare se è il caso di fare una sosta. Il sistema genera una serie di messaggi e segnalazioni acustiche in concomitanza con una serie di eventi specifici (partenza, acquisizione segnale GPS, superamento limite di velocità, ecc.). I messaggi vocali sono in lingua inglese mentre tutte le funzioni sono impostabili, dalla velocità di attivazione del dispositivo LDWS, alla soglia del sensore di movimento.

Caratteristiche modulo base: alimentazione 12 Vdc; consumo 300 mA; temperatura di lavoro -20°C + 70°C; dimensioni 74 x 139 x 32mm; peso 120 g.

Caratteristiche telecamera: alimentazione 12 Vdc; consumo max 100mA; temperatura di lavoro -30°C + 80°C; dimensioni 71,5 x 67 x 39mm; peso 50g.

FR340 (set completo) — **€ 499,00**

Codice MIP 267079

Via Adige, 11 - 21013 Gallarate (VA) • Tel. 0331/799775 - Fax. 0331/778112

Disponibili presso i nostri Rivenditori o nel nostro punto vendita di Gallarate (VA). Caratteristiche tecniche e vendita on-line direttamente sul sito *www.futurashop.it*

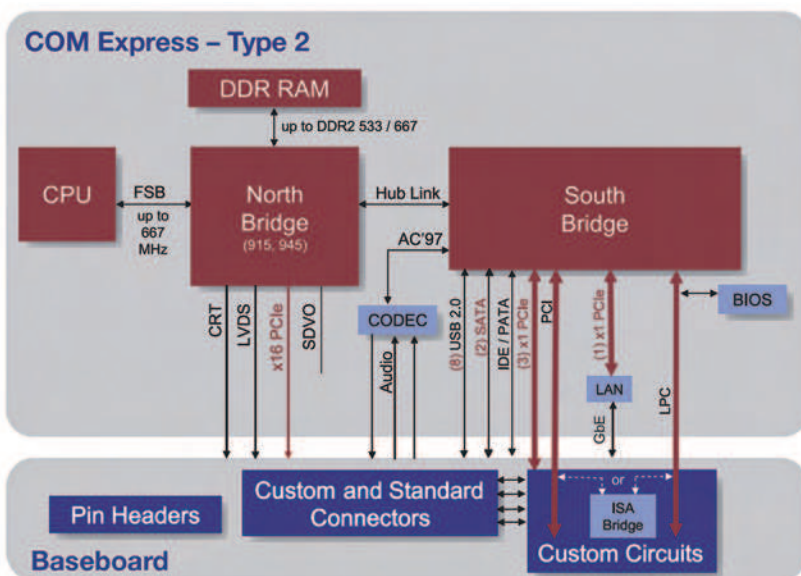



Figura 12
Schema a blocchi di una scheda COMExpress

di soluzioni custom, in tutti quei casi in cui i tradizionali single board computer non risultano adatti, sia per ragioni meccaniche sia per la mancanza di espandibilità.

Riassumendo, un COM si può definire un modulo che integra al proprio interno tutti i componenti necessari per implementare le funzioni di un computer host "bootable", reso disponibile sotto forma di dispositivo off-the-shelf.
Le espansioni del sistema e la sua personalizzazione vengono implementate su una carrier board. Insieme COM e carrier board costituiscono un SBC.

Novità del PCI Express

Il PCI Express è l'evoluzione del bus di espansione PCI (Peripheral Component Interconnect), introdotto con i primi Pentium che sta anche prendendo il posto della vecchia interfaccia per schede grafiche, l'AGP.

Figura 14
Logo del PCI Express

È basato su un trasferimento dei dati seriale, a differenza di quello parallelo del PCI, che semplifica il layout del PCB delle schede madri ed è costituito da una serie di canali. Ciò consente una notevole modularità, in quanto possono essere aggregati più canali per aumentare la banda passante disponibile o per supportare particolari configurazioni come l'utilizzo di due o più schede video. Inoltre la bandwidth di ciascun canale è indipendente da quella degli altri.

Anche se il nome è lo stesso, il PCI Express si divide in due tipi:

- PCI x1, costituito da un singolo canale, che offre una bandwidth full duplex di 266 MBytes/sec e che diventerà il sostituto "ufficiale" del vecchio standard PCI. Una caratteristica curiosa sono le dimensioni, nettamente inferiori rispetto a quelle dello standard PCI (come mostrato nella foto) e la possibilità (almeno in teoria) di praticare l'hot swap, ovvero di poter sostituire una scheda a PC acceso (ora impossibile per lo standard PCI ma già disponibile con SATA e USB.)
- PCI x16, costituito da 16 canali, potendo offrire così il doppio della velocità rispetto allo standard AGP 8x, teoricamente in grado di trasferire 2,5 Gb/s sia in entrata che in uscita.

PCI Express è infine progettato per sostenere il sempre maggior fabbisogno energetico delle schede video di ultima generazione. Infatti, a differenza dello slot AGP, in grado di erogare un massimo di 50 Watt, l'attuale revisione di PCIe supporta carichi fino a 75W, permettendo così di eliminare il connettore MOLEX dalle schede di fascia medio-bassa anche se è rimasto per tutte le altre.

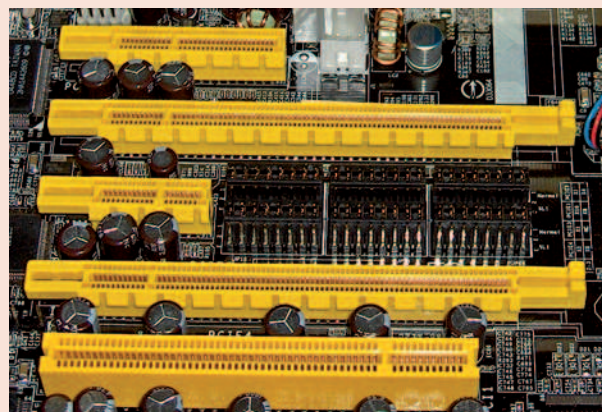



Figura 15
Comparazione degli slot PCI Express e PCI tradizionali. Slot PCI Express (Partendo da sopra: x4, x16, x1 e x16). PCI tradizionale a 32 bit (ultimo in basso)

Riferimenti utili

[1] Specifiche ETX: http://de.kontron.com/downloads/white_papers/ETXSpecV3.02.pdf
[2] it.kontron.com/index.php?id=82&cat=11
[3] www.etx-ig.com/members/members.php
[4] emea.kontron.com/downloads/white_papers/ETXSpecV3.02.pdf
[5] emea.kontron.com/index.php?id=780&cat=31&productid=1322#d3-O
[6] www.trustedcomputinggroup.org
[7] www.congatec.com/x945.html
[8] www.picmg.org
[9] Specifiche XTX: www.xtx-standard.org/fileadmin/documents/XTX-Spec12.pdf
[10] Specifiche COMExpress: www.comexpress-extension.com/datasheet/COM_Express_Extension_Rev007_com_ig.pdf
[11] www.microsoft.com/windows/embedded
[12] www.uclinux.org

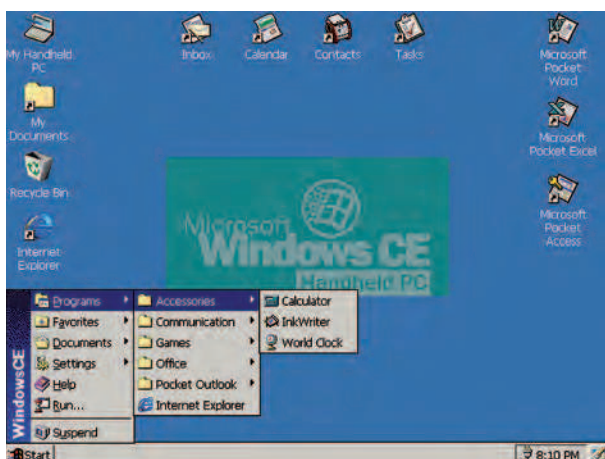



Figura 13
Schermata del sistema operativo Windows Embedded CE

I vantaggi che ne derivano sono evidenti:

- Scalabilità, con la possibilità di espandere la capacità del sistema senza cambiare la baseboard.
- Minore complessità, poiché si tratta di un sistema già testato e collaudato.
- Riduzione dei tempi di progettazione.

More Info Please!
Inserisci il Codice 267072 alla pagina
www.farelettronica.com/mip
oppure utilizza il **modulo** a pagina **3**

Prima parte: *Giugno 2007*
Struttura e risorse interne

Seconda parte: *Luglio/Agosto 2007*
L'utilizzo e le caratteristiche del sistema di sviluppo

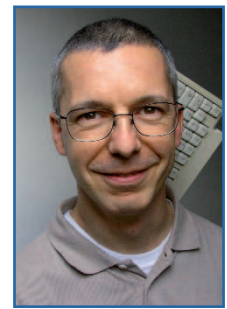
Terza parte: *Settembre 2007*
Programmi demo ed applicazioni pratiche

Microprocessori MSP430

In questa ultima puntata verrà descritta tutta una serie di programmi per evidenziare le caratteristiche delle periferiche utilizzate. Ogni programma è un esempio essenziale di come utilizzare la specifica risorsa del microprocessore e l'insieme dei programmi costituisce una base con la quale costruire la propria applicazione. Tutti i programmi sono in linguaggio C ma saranno dati alcuni cenni sull'assembler della famiglia MSP430 al fine di evidenziare le potenzialità di questo linguaggio macchina.

Tutti i programmi descritti nel seguente articolo sono stati sviluppati prendendo spunto da alcuni esempi presenti nei vari siti internet dedicati all'argomento e sono qui descritti a puro scopo didattico. È comunque disponibile in rete una grande varietà di esempi più completi e sofisticati anche grazie al fatto che le versioni più vecchie di microprocessore della famiglia degli MSP430 hanno più di dieci anni. Per avere il codice sorgente dei programmi è possibile effettuate il download del file Raccolta di Esempi.ZIP che contiene tutti gli esempi descritti, coordinati in un unico progetto per l'IDE IAR, direttamente dal sito di questa rivista. Oppure, data la brevità della maggior parte dei programmi, è possibile copiarli direttamente dalle pagine dell'articolo. Nel caso del download, è disponibile il file di workspace o progetto che imposta l'IDE secondo quanto richiesto dalle singole applicazioni. Nel caso di programmi copiati dalla rivista oppure da altra fonte, è consigliato creare un progetto per ogni esempio in modo da conservare le specifiche impostazioni dell'IDE per eventuali usi futuri. Le impostazioni sono molto simili per tutti i programmi e possono essere riassunte nei seguenti passi:

- Inserire la chiavetta col sistema di sviluppo nella presa USB del vostro PC.
- Lanciare l'IDE con Start > Programmi > IAR Systems > IAR Embedded Workbench KickStart for MSP430 V3 > IAR Embedded Workbench.
- Nel caso che l'IDE si apra come di consueto sull'ultimo progetto utilizzato, dal menù File, selezionare Close Workspace.
- Creare quindi un nuovo workspace dal menù File selezionando la voce New > Workspace.
- Creare un nuovo progetto dal menù Project > Create New Project. Selezionare ovviamente nome e locazione del progetto.
- Nella finestra del workspace, selezionare il nuovo progetto con il tasto destro del mouse e selezionare Options ovvero le opzioni che personalizzeranno questo nuovo progetto.
- Nel folder Target selezionare il microprocessore per cui stiamo sviluppando il progetto che nel nostro caso, volendo verificarlo sulla target board del nostro sistema di sviluppo, sarà MSP430F2013. A riguardo delle opzioni generali non modificare quelle che sono le impostazioni di default.
- Nel campo Linker aggiungere alle opzioni esistenti la generazione di informazioni extra per il C-SPY selezionando Allow C-SPY specific extra output file.
- Restando nella finestra delle opzioni, scegliere il campo Debugger e nell'apposito campo scegliere se si vuole usare come strumento di debug il simulator o il FET debugger. Nel nostro caso, data la disponibilità dell'HW, possiamo permetterci di scegliere FET debugger ma anche il simulatore merita di "farci un giro". Selezionare inoltre Override default lasciando comunque nella finestra che si è abilitata, la scelta proposta dal sistema che fa riferimento al microprocessore in uso MSP430F2013.
- Se si è scelto FET debugger, occorre personaliz-

Programmi demo ed applicazioni pratiche

di Roberto Verzino

zare le impostazioni e quindi selezionare la voce FET debugger tra le due presenti sotto alla voce debugger. In questo folder specificare che il nostro debugger è su porta USB, e che desideriamo che il download sia verificato selezionando verify download. In questo folder si possono scegliere alcune interessanti opzioni che potranno tornarci utili magari con programmi più avanzati: Si può scegliere infatti di non fare il download del codice sul microprocessore ma solo di connettersi per verificare l'esecuzione del codice già presente, si può scegliere di cancellare, in fase di download, solo la memoria codice o anche l'information memory ed anche di connettersi ad un target che sta funzionando per quelle operazioni di ispezione che sono consentite su un target che sta funzionando.

- Selezionare con il tasto destro, nella finestra di workspace, il nome del nostro progetto e scegliere Add > Add Files ed aggiungere il file sorgente prescelto. Una volta che il file è comparso nella finestra di workspace è sufficiente fare doppio click sul suo nome per avviarne l'editing.
- Attivare il Make del file dal menù Project o con l'apposito tasto sulla barra dei comandi. Se non ci sono errori segnalati nella finestra di log della compilazione, possiamo fare il download del programma selezionando Debug sempre dal menù Project o con l'apposito tasto sulla barra dei comandi.
- Si attiva così l'ambiente di debug che ci consente di avviare l'esecuzione del programma, fermarlo, inserire breakpoints e così via, il tutto tenendo sotto controllo il codice sorgente C, l'assembler corrispondente e qualunque altra informazione ci possa interessare. Nella videata di figura 1 è possibile vedere una piccola parte delle informazioni che il debugger può fornire durante la sua attività.

Per l'attivazione e lo sfruttamento di tutte le potenzialità di questo tool, si consiglia di consultare la documentazione allegata nel CD di installazione in quanto le possibilità sono davvero enormi ma non tutte sono sempre necessarie. Le principali sono:

- L'uso dei breakpoint che si attivano o disattivano con il doppio click sulla riga verticale grigia alla sinistra della finestra del codice sorgente, all'altezza del punto prescelto.
- L'aggiunta di una variabile nella finestra di watch (attivabile dal menù View) effettuabile con un singolo click nella prima locazione disponibile nella finestra e digitando il nome della variabile da monitorare.
- Il controllo dei registri interni del microprocessore effettuabile attivando la finestra register (attivabile dal menù View).

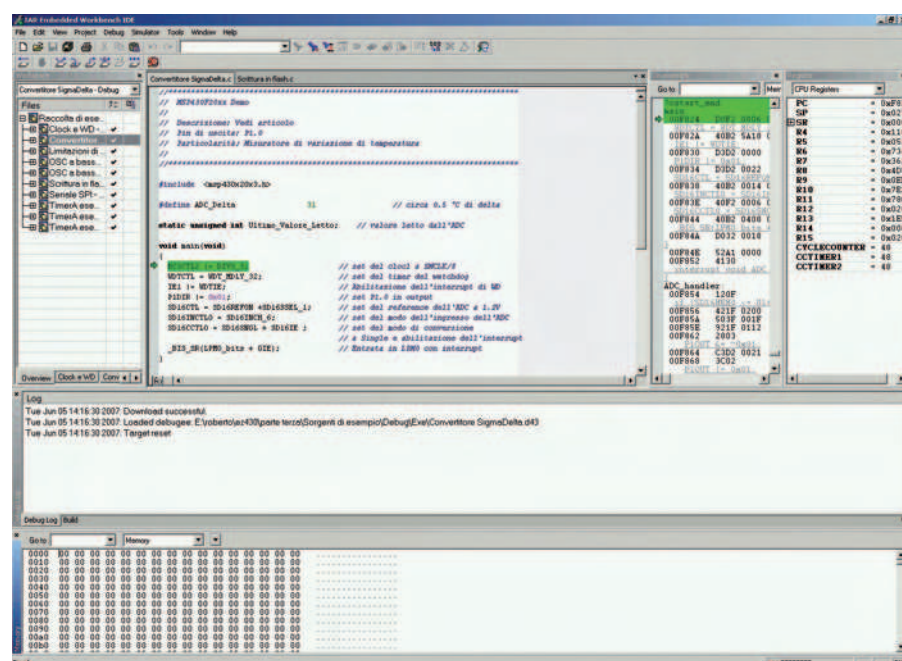



Figura 1
Videata con il debugger in azione

Listato 1

```
//*******************************
//
//  Descrizione: vedere articolo
//  Pin di uscita: P1.0 con una
//  quadra di 1/10 di duty cycle
//  Particolarità: funzionamento
//  a bassissima freq. ~ 1.5kHz
//
//*******************************

#include  <msp430x20x3.h>

void main(void)
{
1 volatile unsigned int i;
2 WDTCTL = WDTPW + WDTHOLD;
3 BCSCTL3 |= LFXT1S_2;
4 IFG1 &= ~OFIFG;
5 __bis_SR_register(SCG1 + SCG0);
6 BCSCTL2 |= SELM_3 + DIVM_3;
7 P1DIR = 0xFF;
8 P1OUT = 0;
9 P2DIR = 0xFF;
10 P2OUT = 0;


11 for (;;)
12 {
13   P1OUT |= 0x01;
14   for (i = 10; i > 0; i–);
15   P1OUT &= ~0x01;
16   for (i = 100; i > 0; i–);
17 }
}
```

Descrizione dei programmi

Nei successivi paragrafi viene fornita una descrizione dei programmi scelti per illustrare le caratteristiche del microprocessore della target board, le potenzialità dell'IDE e del compilatore C. Sono stati scelti principalmente programmi che evidenziano il loro funzionamento mediante il led montato sulla target board ma è ovvio che le possibilità si moltiplicano se si ammette di montare un interruttore collegato ad un pin di input del microprocessore oppure un dispositivo seriale (una memoria, un convertitore D/A, un altro microprocessore) collegato alla linea seriale SPI o I2C che sarà descritta in seguito. I commenti presenti in ogni riga dei listati originali (scaricabili dal sito della rivista) sono stati omessi nella versione stampata per esigenze grafiche.

OSC a bassa frequenza.c

E' un programma nel quale l'uso del pin di output per accendere il led, è subordinato all'uso di una delle modalità di low power disponibili in questo microprocessore. Una funzione banale come accendere il led è stata quindi implementata generando un segnale con un duty cycle che ha una fase attiva del dieci per cento derivata dal funzionamento del microprocessore ad una frequenza molto bassa (circa 1,5 Khz).
Alla riga 1 viene dichiarata una variabile che viene dichiarata volatile per evitere che l'ottimizzazione fatta dal compilatore possa rimuoverla e rendere il debug meno facile da seguire. Nella riga 2 viene fermato il watch dog per evitare che intervenga. Nelle righe 3,4 e 5 vengono settate le opzioni che permettono al microprocessore di funzionare senza oscillatore esterno e nemmeno il generatore di clock DCO è attivato bensì è utilizzato solo un oscillatore interno a basso consumo che tramite opportuni divisori fornisce la frequenza di circa 1,5KHz. Nelle righe 6,7,8, 9 e 10 viene definita la direzione e l'uso dei pin di IO attinenti alle porte logiche 1 e 2. Nelle righe da 11 a 17 c'è il ciclo for che accende il led, attende il decremento della variabile e spegne il led. Attende nuovamente il decremento della variabile che questa volta vale 10 volte il valore precedente per generare il duty di circa 1/10. Le modifiche piu semplici possono essere quelle di variare il duty cycle per rendere più visibile il lampeggio del led e/o la frequenza di lavoro del microprocessore (listati 1).
Per avere una indicazione della efficienza dell'assembler di questa famiglia di microprocessori, segue lo stesso programma appena illustrato ma scritto in linguaggio macchina (listato 2).
Come si può notare, la dimensione dei listati è simile ed anche le istruzioni. L'occupazione di memoria infatti è per la versione in C di 91 byte e per la versione assembler di 74 byte.

Scrittura in flash.c

Questo esempio è l'illustrazione di una delle principali caratteristiche della famiglia dei microprocessori MSP430FXXXX: la possibilità di accedere in scrittura alla propria memoria codice. In questo particolare esempio la scrittura viene effettuata in due segmenti di memoria chiamati "information memory" e quindi deputati alla memorizzazione di parametri e dati variabili da ritenere anche in assenza di alimentazione. La potenzialità è comunque quella di poter accedere a qualunque area e di gestire le protezioni dei segmenti. Ovviamente non si tratta di memoria ram quindi gli algoritmi di accesso sono stati ridotti al minimo pur conservando un livello di sicurezza tale da proteggere il sistema da accessi non intenzionali. Come indicato nel

Listato 2

```
;********************************
;
; Toggle di P1.0 con MCLK = VLO/8
;
;   Descrizione: vedere articolo
;   Pin di uscita: P1.0 con una
;   quadra di 1/100 di duty cycle
;   Particolarità: funzionamento
;   a bassissima freq.   ~ 1.5kHz
;
;********************************
#include  "msp430x20x3.h"
;----------------
            ORG      0F800h
;----------------
RESET
   mov.w    #0280h,SP
   mov.w    #WDTPW+WDTHOLD,&WDTCTL
   bis.b    #LFXT1S_2,&BCSCTL3
   bic.b    #OFIFG,&IFG1
   bis.w    #SCG1+SCG0,SR
   bis.b    #SELM_3+DIVM_3,&BCSCTL2
   mov.b    #0FFh,&P1DIR
   clr.b    &P1OUT
   mov.b    #0FFh,&P2DIR
   clr.b    &P2OUT
loop
   bis.b    #001h,&P1OUT
   mov.w    #020,R15
DLY1
   dec.w    R15
   jnz      DLY1
   bic.b    #001h,&P1OUT
   mov.w    #02000,R15
DLY2
   dec.w    R15
   jnz      DLY2
   jmp      loop

;----------------
;           Interrupt Vectors
;----------------
            ORG      0FFFEh
            DW       RESET
            END
```

sorgente, l'esecuzione del programma prevede un ciclo continuo di ri-scritture della flash; dato che ciò avviene in tempi molto contenuti, in fase di debug occorre mettere un breakpoint sull'istruzione NOP indicata (quello all'interno del while) al fine di non far riscrivere troppe volte la flash che sopporta un numero non infinito di scritture ("solo" 100.000 cicli).

Nel main possiamo osservare l'inizializzazione del clock per il processore e per la macchina di gestione di accesso alla flash. Viene poi chiamata all'infinito la routine di scrittura in flash e la copia del segmento C nel segmento D. La prima delle funzioni chiamate, la Scrittura_Seg_C, implementa la procedura di accesso alla flash che è ampiamente descritta nella documentazione della famiglia del microprocessore ma che possiamo riassumere nelle seguenti fasi:

- Set del bit di erase nel registro di controllo accesso alla flash per richiedere una operazione di cancellazione. Questo set è fatto con la contemporanea scrittura di una chiave di accesso di sicurezza detta FWKEY che permette di evitare che il comando sia scritto per sbaglio.
- Cancellazione del bit di protezione del settore riconfermando la FWKEY.
- Una scrittura fittizia per attivare la macchina di scrittura
- In questo modo il settore indirizzato è stato cancellato e si può direttamente passare alla scrittura dei dati attivando il bit di write (sempre confermato dalla FWKEY) che predispone la macchina di gestione flash all'accesso.
- La scrittura vera e propria dei dati che nel nostro caso sono i 64 byte del settore posti al valore di "value".
- Cancellazione del bit di write per chiudere correttamente la sequenza di scrittura.
- Conferma del bit di lock della flash comprensiva della FWKEY.

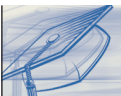
La funzione Copia_Seg_C_in_D implementa lo stesso algoritmo ma invece di scrivere il valore "value", nel settore D copia il contenuto del settore C dimostrando come sia possibile accedere a memoria codice e memoria dati nello stesso ciclo "for" sia in lettura che in scrittura. Il programma è chiaramente dimostrativo e va usato con il breakpoint del quale si è già parlato, in modo da poter verificare l'avanzamento delle varie fasi (listato 3).

Convertitore SigmaDelta.c

In questo programma si comincia a fare un uso contemporaneo di più risorse del microprocessore. Il convertitore analogico/digitale sigma-delta "legge" un canale analogico connesso al sensore di temperatura interno al microprocessore campionandolo ogni 240ms. Il valore rilevato viene comparato con la precedente lettura e se la differenza è maggiore di 0,5 gradi centigradi (circa) il led connesso alla porta P1 viene acceso altrimenti viene spento. Per generare la tempistica del cam-

Listato 3

```
///******************************
//   MSP430F20xx Demo
//
//   Descrizione: Vedi articolo
//   Pin di uscita: nessuno
//   Particolarità; Scrittura in
//   flash     (information memory)
//
//******************************

#include  <msp430x20x3.h>

char  data;

// *** Prototipi ***
void Scrittura_Seg_C (char data);
void Copia_Seg_C_in_D (void);

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  BCSCTL1 = CALBC1_1MHZ;
  DCOCTL = CALDCO_1MHZ;
  FCTL2 = FWKEY + FSSEL0 + FN1;
  data = 0;
  while(1)
  {
    Scrittura_Seg_C(data++);
    Copia_Seg_C_in_D();
    _NOP();
  }
}

void Scrittura_Seg_C (char data)
{
  char *Flash_ptr;
  unsigned int i;

  Flash_ptr = (char *) 0x1040;

  FCTL1 = FWKEY + ERASE;
  FCTL3 = FWKEY;
  *Flash_ptr = 0;

  FCTL1 = FWKEY + WRT;
  for (i=0; i<64; i++)
  {
    *Flash_ptr++ = data;
  }
  FCTL1 = FWKEY;
  FCTL3 = FWKEY + LOCK;
}

void Copia_Seg_C_in_D (void)
{
  char *Flash_ptrC;
  char *Flash_ptrD;
  unsigned int i;

  Flash_ptrC = (char *) 0x1040;
  Flash_ptrD = (char *) 0x1000;
  FCTL1 = FWKEY + ERASE;
  FCTL3 = FWKEY;
  *Flash_ptrD = 0;

  FCTL1 = FWKEY + WRT;

  for (i=0; i<64; i++)
  {
   *Flash_ptrD++ = *Flash_ptrC++;
  }

  FCTL1 = FWKEY;
  FCTL3 = FWKEY + LOCK;
}
```

pionamento è utilizzato il watchdog e la ISR è usata per dare lo start al convertitore per la prossima lettura. Questo banale rivelatore di variazioni di temperatura suggerisce interessanti progetti quali controllo di carica-batterie, termostati e sicurezze varie su circuiti di potenza. Vediamo più nel dettaglio il compattissimo sorgente.
La define ADC_Delta rappresenta il valore decimale corrispondente a mezzo grado Celsius nella scala di lettura del convertitore, date le impostazioni prescelte. Segue la definizione della variabile usata per la memorizzazione del valore precedente. Vi sono poi sette righe di impostazioni nelle quali viene attivato il watch dog ed abilitato l'interrupt ad esso associato, attivata la porta di output e programmato il modo di funzionamento del convertitore analogico/digitale. Il processore viene quindi mandato in modalità low power LPM0 in attesa di interrupts. Con la direttiva #pragma vector=SD16_VECTOR informo il compilatore che la seguente funzione è l'handler di gestione dell'interrupt che ha come vettore SD16_VECTOR. In questo modo il compilatore metterà automaticamente, all'indirizzo del vettore specificato un salto alla funzione indicata. Il tipo __interrupt per la funzione void SD16ISR(void) specifica che si tratta di un handler di interrupt e quindi l'istruzione finale dell'assembler corrispondente non deve essere

un RET (ritorno da call) ma una RETI (ritorno da handler di interrupt. Nel corpo della funzione che sarà attivata dall'interrupt di fine conversione del convertitore, c'è l'analisi della lettura effettuata e la conseguente azione sul led. A questo punto il convertitore attiverà un altro ciclo di conversione solo quando l'handler dell'interrupt del watch dog provvederà a dare un nuovo segnale di start. Il debug di questo programma è molto semplice e "curioso": dopo aver dato il run al programma, sarà sufficiente alitare sulla vostra target board per vedere accendere il led (a meno che non stiate debuggando a 37°C ...) (listato 4).

Listato 4

```
///*******************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di uscita: P1.0
//  Particolarità; Misuratore di
//  variazione di temperatura
//
//*******************************

#include  <msp430x20x3.h>
#define ADC_Delta  31 // ~0.5 °C

// Var. per Ultimo Valore Letto
static unsigned int     U_V_L;

void main(void)
{
  BCSCTL2 |= DIVS_3;
  WDTCTL = WDT_MDLY_32;
  IE1 |= WDTIE;
  P1DIR |= 0x01;
  SD16CTL = SD16REFON+SD16SSEL_1;
  SD16INCTL0 = SD16INCH_6;
  SD16CCTL0 = SD16SNGL + SD16IE ;

  _BIS_SR(LPM0_bits + GIE);
}

// Handler interrupt dell'ADC
#pragma vector=SD16_VECTOR
__interrupt void ADC_handl(void)
{
 if(SD16MEM0 <= UVL + ADC_Delta)
  P1OUT &= ~0x01;
 else
  P1OUT |= 0x01;
 Ultimo_Valore_Letto = SD16MEM0;
}

// Handler interrupt del watchdog
#pragma vector=WDT_VECTOR
__interrupt void wtchdg_tim(void)
{
  SD16CCTL0 |= SD16SC;
}
```

TimerA esempio 1.c

Questo programma usa 2 unità di Capture Compare Register (CCRx) e l'overflow del Timer_A e per generare quattro onde quadre a frequenze differenti. Per evidenziare i comportamenti dei CCR, è stata scelta una modalità che prevede il toggle del pin di uscita corrispondente a ogni CCR (P1.1 per CCR0 e P1.2 per CCR1). In questo modo i pin pilotati dai CCRX cambiano stato ogni volta che il conteggio del TAR è uguale al valore impostato nel CCRX stesso. Contemporaneamente al toggle viene generato un interrupt nell'handler del quale viene caricato il conteggio da raggiungere per il prossimo toggle. L'handler dell'interrupt di overflow del Timer_A1 è usato per effettuare il toggle della porta P1.0 mediante apposita istruzione. Date le condizioni di clock impostate (ACLK = n/a, MCLK = SMCLK = TACLK = default DCO circa a 1MHz), su P1.1 sarà presente una forma d'onda a circa 2500Hz, su P1.2 una a circa 500Hz e su P1.0 una a circa 8Hz.

Nel programma è facile individuare le seguenti operazioni:

- Stop del watch dog.
- Collegamento delle porte P1. e P1.2 alle periferiche interne del micro (i CCR).
- Set della direzione dello porte da P1.0 a P1.2 in output.
- Set del modo di funzionamento dei due CCR accedendo al rispettivo registro di controllo CCTLx.
- Selezione del modo di funzionamento del Timer_A ed ingresso in LPM0 con risveglio da interrupt.

Negli handler degli interrupt troviamo una gestione canonica per l'interrupt del Timer_A0 mentre per il timer A2 non è così. La gestione dell'interrupt del Timer_A1 evidenzia una interessante caratteristica di questa architettura ovvero la vettorizzazione della sorgente dell'interrupt: la prima istruzione dell'handler è uno switch sul simbolo TAIV che contiene il vettore specifico associato alla sorgente di quell'interrupt. Così, nello stesso handler ho la gestione (discriminata dal valore di TAIV) dell'interrupt causato dal CCR1 e la gestione dell'interrupt causato dall'overflow del timer (Listato 5).

Listato 5

```
//********************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di out: P1.0, P1.1, P1.2
//  Particolarità; Generazione di
//  3 segnali a freq. Differente
//
//  P1.1 =  ~2500Hz
//  P1.2 =  ~500Hz
//  P1.0 =  ~8Hz
//
//********************************
#include  <msp430x20x3.h>
void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  P1SEL |= 0x06;

  P1DIR |= 0x07;
  CCTL0 = OUTMOD_4 + CCIE;
  CCTL1 = OUTMOD_4 + CCIE;
  TACTL = TASSEL_2 + MC_2 + TAIE;

  _BIS_SR(LPM0_bits + GIE);
}

// Handler interrupt del Timer A0
#pragma vector=TIMERA0_VECTOR
__interrupt void Timer_A0 (void)
{
  CCR0 += 200;
}

// Handler interrupt del Timer A0
#pragma vector=TIMERA1_VECTOR
__interrupt void Timer_A1(void)
{
  switch( TAIV )
  {
    // Int generato da CCR1
    case  2: CCR1 += 1000;
             break;

    // Int overflow del timer
    case 10: P1OUT ^= 0x01;
             break;

 }
}
```

TimerA esempio 2.c

In questo esempio viene implementato il toggle della porta P1.1 (purtroppo non quella del led) utilizzando una modalità associata al funzionamento del timer A. Questa modalità prevede che CCR0 definisca il periodo di conteggio del contatore e che quando viene raggiunto il limite impostato, TA0 effettua il toggle del pin ad esso collegato. In questo modo è possibile generare una onda quadra con duty cycle 50%, di frequenza selezionabile mediante CCR0 (in questo caso la frequenza è TACLK/1000), senza usare alcuna risorsa software o tempo di elaborazione CPU. Nel sorgente possiamo facilmente individuare lo stop del watchdog, la selezione dell'uso della porta P1.1 che è connessa ad una periferica interna, la programmazione del CCR0, del timer, ed infine la cosa più interessante: se la cpu non serve, possiamo anche spegnerla (ultima riga) con un _BIS_SR(CPUOFF)! (Listato 6).

TimerA esempio 3.c

Un altro esempio di generazione di forme d'onda è costituito dal sorgente sottostante che implementa la generazione di un segnale PWM (dove la variazione del duty cycle è da effettuarsi modificando i parametri di impostazione dei CCRX) sempre senza l'uso di tempo CPU. In questo caso il timer A è programmato per funzionare in up/down mode. Il registro valore scritto nel CCR0

Listato 6

```
///********************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di uscita: P1.1
//  Particolarità; Generatore di
//  onda senza l'uso di tempo CPU
//********************************
#include  <msp430x20x3.h>

void main(void)
{

  WDTCTL = WDTPW + WDTHOLD;
  P1DIR |= 0x02;
  P1SEL |= 0x02;
  CCTL0 = OUTMOD_4;
  CCR0 = 500-1;
  TACTL = TASSEL_2 + MC_1;

  _BIS_SR(CPUOFF);
}
```

Listato 7

```
//*******************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di uscita: P1.2
//  Particolarità; PWM senza uso
//                di tempo CPU
//*******************************
#include  <msp430x20x3.h>

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  P1DIR |= 0x0C;
  P1SEL |= 0x0C;
  CCR0 = 128;
  CCTL1 = OUTMOD_6;
  CCR1 = 32;
  TACTL = TASSEL_2 + MC_3;
  _BIS_SR(LPM0_bits);
}
```

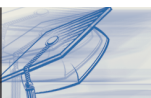
definisce la frequenza del segnale (in particolare il valore del periodo diviso 2) ed il valore scritto in CCR1, il duty cycle. Con i valori forniti nell'esempio, otterremo un segnale con un duty del 75%. Anche in questo caso data che la CPU non interviene nella generazione del segnale, l'ultima riga del listato fa entrare in low power (LPM0) il sistema (Listato 7).

Seriale SPI.c

Questo è l'unico esempio di quelli riportati in questo articolo che non è facilmente verificabile perché implica l'uso di due microprocessori. La soluzione ottimale potrebbe essere quella di farsi prestare da un amico il suo EZ430, saldare 5 fili che uniscano le due target board (3 di linea SPI, uno di reset e uno di massa). In mancanza di tutto questo si può comprare o realizzare un'altra target board ma occorre pensare anche all'alimentazione della board aggiuntiva. In ogni caso è interessante anche la sola analisi dei sorgenti per vedere come sia semplice l'implementazione di una seriale SPI lato master. La descrizione si limita al lato master perché è quello che si deve implementare per accedere alla maggior parte dei dispositivi seriali quali memorie EEPROM, flash etc. Ovviamente, tra i programmi demo che si possono trovare su internet non mancano quelli per l'implementazione del lato slave e per la realizzazione dell'altro diffusissimo standard seriale che è l'I2C. Il programma seguente prevede di inviare un mes-

Listato 8

```
//*******************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di uscita: P1.5/6/7 linea
//  SPI. P1.0 led. P1.4 pin input
//  Particolarità; Seriale SPI
//              modalità master.
//
//*******************************
#include <msp430x20x3.h>


void main(void)
{
  volatile unsigned int i;

  WDTCTL = WDTPW + WDTHOLD;
  P1REN |= 0x10;
  P1OUT =  0x10;
  P1DIR = 0x01;
  USICTL0 |= USIPE7 +  USIPE6 +
             USIPE5 + USIMST +
             USIOE;


  USICTL1 |= USIIE;
  USICKCTL = USIDIV_4 +USISSEL_2;
  USICTL0 &= ~USISWRST;
  USISRL = P1IN;
  P1DIR |= 0x04;
  P1DIR &= ~0x04;
  for (i = 0xFFF; i > 0; i—);
  USICNT = 8;
  _BIS_SR(LPM0_bits + GIE);
}

// Handler interrupt di seriale
#pragma vector=USI_VECTOR
__interrupt void USI_hndler(void)
{
  if (0x10 & USISRL)
    P1OUT |= 0x01;
  else
    P1OUT &= ~0x01;

  USISRL = P1IN;

  USICNT = 8;
}
```

Listato 9

```
//********************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di uscita: P1.0
//  Particolarità; Toggle del led
//   per intervento del watch dog
//
//********************************
#include  <msp430x20x3.h>

void main(void)
{
  P1DIR |= 0x01;
  P1OUT ^= 0x01;
  _BIS_SR(LPM4_bits);
}
```

saggio che rispecchia lo stato del pin di input P1.4. Questo messaggio è inviato continuamente mentre lo stato del led rappresenta il livello logico del pin di input remoto. Nel listato sono individuabili le linee di codice per effettuare:

- Il blocco del watch dog.
- Il set della direzione delle porte.
- L'abilitazione del pull-up interno sul pin P1.4 per la sua gestione come input (normalmente a 1 logico, se chiuso a massa va a 0).
- Le operazioni di inizializzazione della seriale fino alla scrittura del dato da inviare (USISRL = P1IN;).
- La generazione di un breve impulso di reset del dispositivo slave effettuata portando a 1 logico e poi a 0 il pin P1.2. La particolarità di questa operazione è che viene effettuata non con la scrittura di un valore logico su di un pin già configurato come input ma girandolo temporaneamente in output. Questo forza il livello logico del pin mentre quando ritorna in input il livello logico è quello imposto dall'esterno ovvero dal pull-up presente sul segnale.
- La generazione di un piccolo ritardo SW per attendere il completamento delle operazioni di reset dello slave imposte con la transizione descritta al punto precedente.
- L'attivazione della trasmissione viene effettuata ponendo a 8 il valore del contatore di bit trasmessi. In questa macchina infatti, il contatore dei bit trasmessi (o da trasmettere), è l'entità che genera l'interrupt di fine "buffer empty" quando arriva a zero.
- Segue poi l'handler di gestione dell'interrupt della seriale che nelle prime istruzioni si premura di rappresentare sul pin P1.4 lo stato logico rilevato nel buffer di ricezione (ovvero lo stato del pin del dispositivo remoto) e poi di caricare nel buffer di trasmissione il valore del pin P1.4 locale (Listato 8).

Clock e WD.c

In questo semplice esempio è rappresentata una condizione di fail che provoca l'attivazione del watch dog ma la particolarità sta nel fatto che il codice stesso vuole simulare una disattivazione accidentale del watch dog. Così il sorgente, composto di 3 righe esegue:

- Come visibile il watch dog questa volta non viene fermato ma viene girata in output la porta del led.
- Quindi viene effettuato il toggle del pin P1.0.
- Pio vengono disabilitati tutti i clock del processore che in questo modo si pone in una condizione simile al LPM4. A questo punto, avendo disabilitato tutti i clock della macchina, il timeout watch dog scatta ugualmente e resetta il microprocessore che ricomincia l'esecuzione del codice. Il led in questo modo lampeggerà continuamente e l'operazione di disabilitazione di tutti i clock non ha inibito il corretto funzionamento del watch dog (Listato 9).

Listato 10

```
//********************************
//  MSP430F20xx Demo
//
//  Descrizione: Vedi articolo
//  Pin di uscita: P1.0
//  Particolarità; Reset del
//  processore su fetch illegale
//
//********************************
#include  <msp430x20x3.h>

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  P1DIR |= 0x01;
  TAR = 0x3FFF;

  for (;;)
  {
    volatile unsigned int i;
    P1OUT ^= 0x01;
    i = 50000;
    do (i–);
    while (i != 0);

    ((void (*)())0x170)();
  }
}
```

Limitazioni di fetch.c

Questo esempio evidenzia una delle particolarità di questo microprocessore ovvero il controllo dell'indirizzo dal quale viene prelevato il codice da eseguire. Dato che l'architettura Von Neumann consente il fetch di codice anche dalla ram e che le periferiche sono memory mapped, per limitare le possibilità di errore è stato inserito un controllo apposito sull'address del codice in corso di esecuzione. Se il fetch viene effettuato in una area illegale (ad esempio dagli indirizzi delle periferiche) il microprocessore viene resettato. In questo caso il toggle del led, effettuato mediante l'applicazione dell'operatore logico XOR sullo stato di P1.0 stesso, rappresenta il reset del microprocessore e quindi l'avvenuto "trap" della condizione illegale generata mediante l'astuta notazione ((void (*)())0x170)(); che corrisponde alla chiamata di una funzione residente all'address 0x170. Altro aspetto particolare di questo programma è costituito dal fatto che il registro TAR (il counter del timer A) è caricato con 0x3FFF all'inizio del programma. La falsa chiamata a funzione di cui abbiamo parlato, eseguirebbe proprio il codice presente nel registro del contatore, codice che corrisponde ad un JMP $ ovvero ad un loop infinito. L'intento del programma è anche quello di dimostrare che altri dispositivi della stessa famiglia, ad esempio MSP430F1XX, non posseggono il trap sul fetch da area illegale e con questo stesso programma, invece di resettarsi, eseguono all'infinito (Listato 10). A questo punto penso possano essere stati presentati un numero sufficiente di esempi e di curiosità tale da mettervi nelle condizioni di realizzare tutto quello che volete con il vostro EZ430. Molto altro codice è disponibile in rete e specialmente nei siti citati tra i riferimenti. Sono stati volutamente scelti codici verificabili senza alcun HW aggiuntivo oltre l'EZ430. Collegando pero' al microprocessore ulteriori dispositivi quali memorie, I/O expander, display LCD e sensori, beh, non ci saranno più limiti alla fantasia. Buon divertimento.

Ringrazio inoltre l'esimio Bruno Soi per la collaborazione offertami.

Riferimenti utili

[1] Sito ufficiale Texas Instruments: www.ti.com
[2] Manuale famiglia MSP430F20XX: SLAS491C.pdf
[3] Manuale EZ430-F2013 user guide: SLAU176A.pdf
[4] Sito di applicazioni e cultura su MSP430: www.eg3.com/msp430.htm
[5] Digressioni varie sul mondo embedded: www.embeddedrelated.com

More Info Please!

Inserisci il Codice **267082** alla pagina **www.farelettronica.com/mip** oppure utilizza il **modulo** a pagina **3**

Prima parte: *Settembre 2007*
Trasduttori di posizione (I)

Seconda parte: *Ottobre 2007*
Trasduttori di posizione (II)

Terza parte: *Novembre 2007*
Trasduttori di posizione (III)

Trasduttori

Impariamo a conoscere e ad usare questi interessanti dispositivi che occupano un posto di tutto rilievo nelle attuali applicazioni elettroniche.

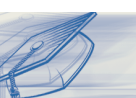
La parola *trasduttore* indica qualunque dispositivo in grado di convertire, secondo sue proprie peculiarità, le variazioni di una grandezza fisica, generalmente non di natura elettrica, in segnali o dati. In altre parole, i trasduttori, ci consentono di controllare e misurare, per mezzo di appositi circuiti, grandezze fisiche quali, ad esempio, luce, umidità, temperatura, ecc. In queste pagine ci occuperemo di un particolare tipo di trasduttore, forse uno dei meno noti, il trasduttore di posizione. Pur essendo una categoria di componenti a noi meno familiare, rispetto, ad esempio, ai trasduttori di temperatura o di segnali acustici, i trasduttori di posizione sono molto presenti fra gli oggetti che accompagnano la nostra vita. Possiamo stabilire, fin d'ora, due categorie fondamentali di trasduttori di posizione: quelli che forniscono grandezze analogiche, dette anche continue, e quelli che forniscono dati digitali, detti anche grandezze discrete o quantizzate. Prima di passare ad una trattazione più dettagliata, possiamo osservare la tabella 1 in cui sono riassunte le principali tipologie di trasduttori con le loro caratteristiche essenziali.
Nella tabella 1 viene anche indicata la risoluzione per ogni tipologia di sensore anche se, data la quantità e qualità dei modelli commercializzati, è stato possibile solo fornire un'indicazione di massima. Oltre alla risoluzione vi sono altri parametri che possono contraddistinguere qualitativamente, e quindi caratterizzare, le varie tipologie di trasduttori di posizione: accuratezza, linearità, isteresi e risposta in frequenza. Prenderemo in considerazione per un breve approfondimento solo questi quattro, tralasciandone altri, meno decisivi, per non appesantire troppo questa trattazione. L'accuratezza, da non confondersi con la precisio-

Il compito di un trasduttore di posizione, ricordiamolo, è convertire lo spostamento di un suo organo, generalmente solidale con un sistema meccanico, in una grandezza elettrica. Di solito lo spostamento si intende su un solo asse, più raramente su due assi (come ad esempio il mouse di un PC), oppure, caso molto comune, rotatorio. Nel primo caso la misura dello spostamento sarà espressa in un'unità metrica, ad esempio millimetri, nel secondo si avrà invece a che fare con i gradi angolari. Facciamo un esempio.
Un potenziometro di tipo commerciale possiede un arco di rotazione di 270° che, lo ricordiamo, è inferiore di ben 90° ad un giro completo. Supponendo che il suo valore sia di 1 KΩ avrà una risoluzione pari a 1000/270, ovvero di 3,7 Ω per ogni grado di rotazione.
Vediamo adesso cosa accadrebbe con un potenziometro, di identico valore, ma del tipo "multigiri". Questi componenti, molto usuali nel campo della strumentazione, sono dotati di una demoltiplica che permette di coprire l'intero valore su 10 giri completi di 360°.
In questo caso avremmo 1000 / (360 x 10) ovvero 0,27 Ω per ogni gradi di rotazione, valore ben superiore al primo esempio.
Si dirà, ed a ragione, che la risoluzione è aumentata. Se la grandezza convertita non fosse una resistenza bensì, come per la maggior parte dei casi, una tensione o corrente continua si potrebbe parlare, ad esempio, di mV/grado. Che, nel caso di misure lineari, diverrebbe mV/mm, V/m, ecc.

di posizione

di Grazia Ancona

TRASDUTTORI DI POSIZIONE							
	Tipo	Movimento	Grandezza in uscita	Risoluzione	Applicazioni	Costo	Note
Digitali	Potenziometro a filo	Angolare (raramente lineare)	Resistenza (in pratica tensione)	Bassa	Controlli manuali di tensione, corrente e segnali AC o in sistemi analogici ove circolano correnti non trascurabili	Basso	Spesso, per migliorare la risoluzione, si usano demoltipliche
	Encoder assoluti	Angolare	Dati a gruppi di n bit	Medio-bassa	Controllo di cinematismi e posizione angolare di motori elettrici su sistemi numerici	Medio-alto	La risoluzione è proporzionale al numero dei bits
	Encoder incrementali	Angolare (raramente lineare)	Impulsi sfasati di 90° e informazioni su giri completi compiuti	Da media ad altissima	Controllo di velocità, accelerazione e senso di rotazione, cinematismi con sistemi di controllo numerici	Medio-alto (basso per i tipi a contatti striscianti)	Rispetto al tipo assoluto richiede una parte elettronica più complessa
	Commutatori codificati	Angolare (raramente lineare)	Dati a gruppi di bit (3-4 al massimo)	Bassa	Controlli manuali e cinematismi a bassa velocità	Basso	Sono, in assoluto, i componenti più economici e meno affidabili. Molto usati nell'elettronica domestica
Analogici	Potenziometri a grafite e magnetici	Angolare o lineare	Resistenza (in pratica tensione)	Teoricamente infinita, in pratica media	Controlli manuali di tensione, corrente e segnali AC, controllo di cinematismi	Basso (alto per i tipi magneti-ci)	Spesso, per migliorare la risoluzione, si usano demoltipliche
	LVDT	Angolare o lineare	Tensioni CA (convertite in tensione)	Da media ad alta	Controllo di cinematismi e velocità di spostamento, accelerometri	Medio-alto	E' uno dei trasduttori più usati, il segnale d'uscita è sempre riconvertito in CC
	Trasduttori magnetoelastici	Lineare	Onde sonore (conver-tite in tensione)	Altissima	Strumentazione scientifica e controllo di cinematismi di precisione	Alto	E' il trasduttore più preciso e sensibile (con costi elevati), il segnale d'uscita è sempre riconvertito in CC o dati
	Syncro e resolver	Angolare	Tensioni CA (convertite in tensione o dati)	Alta	Controllo di motori, cinematismi e strumentazione medico-scientifica.	Medio-alto	Trasduttori molto versatili particolarmente usati in avionica e nel controllo di motori, sono prodotti integrati monolitici per la conversione del segnale CA in dati

Tabella 1
Principali tipologie di trasduttori di posizione

ne (vedi riquadro di approfondimento), esprime la differenza fra valore ideale (esatto) e valore reale (affetto da errore) della grandezza di uscita del trasduttore. Il valore di accuratezza fornito dal costruttore si riferisce, quasi sempre, ad una ben determinata condizione della grandezza da convertire. L'accuratezza si può esprimere in 3 modi: in % sul fondo scala, in % sulla grandezza in uscita (tensione o corrente) e in valore assoluto della grandezza in ingresso (gradi o misure metriche).
La linearità è un parametro che ci indica quanto la caratteristica reale di un traduttore si discosta da quella ideale. Osservando la figura 1, infatti, possiamo notare come la caratteristica ideale di un trasduttore, che possiamo immaginare come l'espressione grafica della sua funzione di trasferimento, sia una retta perfetta.
La pendenza di questa retta, ovvero l'angolo che essa forma con l'asse delle ascisse, dipende dalle caratteristiche del trasduttore (ad esempio il valore resistivo di un potenziometro). Nella stessa figura 1 appare anche la caratteristica reale, mostrata in colore rosso, con andamento decisamente più irregolare anche se, mediamente, mantiene il tracciato della caratteristica ideale. Il costruttore, generalmente, fornisce il massimo scostamento, evidenziato, nel disegno, dal termine Δr. L'isteresi è un problema che interessa tutti quei trasduttori che, come principio costruttivo, si basano su trasformatori AC e flussi magnetici. Questa prerogativa,

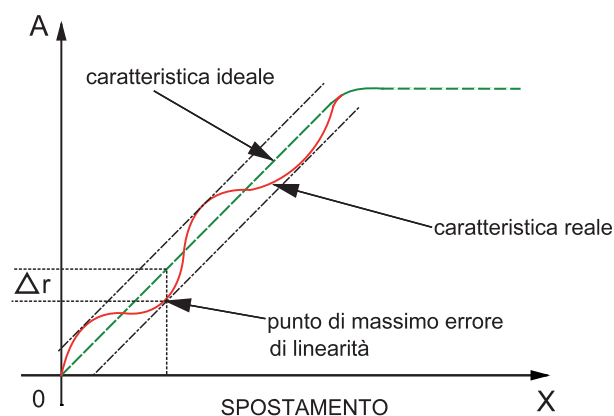
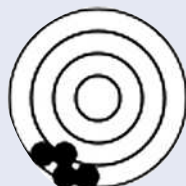



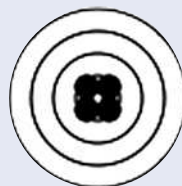
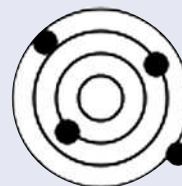
Figura 1
Errore di linearità confrontato con caratteristica ideale

dipendente dalle particolari proprietà dei materiali ferromagnetici, provoca una difformità nella caratteristica di trasferimento in base alla direzione del segnale (crescente o decrescente). Nella figura 2 possiamo avere la dimensione grafica del fenomeno che, per una maggiore evidenza, è stato accentuato. La massima deviazione prodotta dal ciclo d'isteresi sulla grandezza in uscita è evidenziata, come possiamo osservare, da Δ_{max}.

La risposta in frequenza di un trasduttore fa parte di quella categoria di parametri definiti dinamici che, a differenza di quelli finora esaminati, studiano il comportamento del componente quando è

Accuratezza e precisione

Molto spesso si fa confusione fra questi due importanti parametri che, seppure in grossolana semplificazione sembrano indicare concetti abbastanza simili, non sono certo identici. Possiamo definire la precisione come la capacità di ottenere, ad esempio durante una serie di misure, lo stesso risultato senza che ci siano cambiamenti significativi. Infatti, uno dei vocaboli che in inglese traduce questo concetto è: repeatability (ripetibilità). Molto spesso la precisione viene espressa da un intervallo di errore massimo, fornito con la grandezza di misura (esempio: ±0,1 V). L'accuratezza, invece, può essere definita come la capacità di un sistema di misura di avvicinarsi ad un risultato ideale fornito, quest'ultimo, da un sistema campione. I concetti esposti trovano largo impiego in balistica ed è a quest'ultima che ricorreremo per un piccolo esempio grafico che, al di là di ogni formula, dovrebbe chiarire la differenza fra i due parametri. Notate come la distribuzione dei colpi sui bersagli influenzi, in modo diverso, accuratezza e precisione:

Preciso ma non accurato

Accurato ma non preciso

Accurato e preciso

Non accurato e non preciso

È importante tenere presente che il concetto di accuratezza è più importante del concetto di precisione. Infatti, il primo implica necessariamente il secondo ma non viceversa.

Figura 2
Errore d'isteresi confrontato con caratteristica ideale

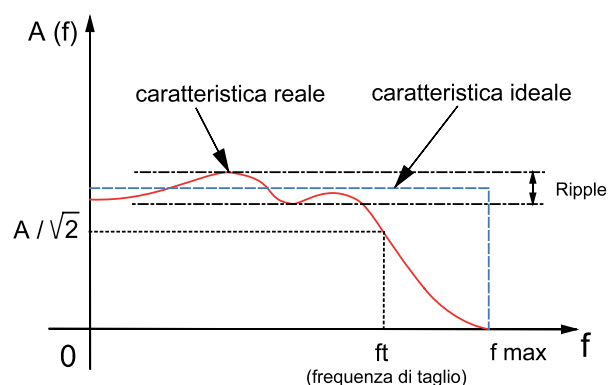



Figura 3
Tipica risposta di un trasduttore nel dominio della frequenza

interessato a segnali che mutano nel tempo molto velocemente. Per fare un esempio, colleghiamo un motore elettrico all'asse di un trasduttore angolare a rotazione continua e, variando la tensione del motore, aumentiamo gradatamente la velocità di quest'ultimo. Fino ad una certa velocità il trasduttore seguirà fedelmente la sua funzione di trasferimento, poi, gradatamente, il suo segnale d'uscita comincerà ad attenuarsi in accordo con l'aumento della velocità del motore. Da questo punto in poi il trasduttore non fornirà più segnali attendibili rispetto alla sua posizione angolare. La tipica risposta in frequenza di un trasduttore, pur variando quantitativamente da modello a modello, è sempre di tipo passa-basso. Ovvero, come è possibile osservare in figura 3, l'aumento della frequenza di funzionamento provoca sempre una diminuzione della grandezza in uscita.
La risposta in frequenza di un trasduttore è influenzata dalla massa delle parti soggette al movimento, che dovrebbero essere le più leggere possibili, dalle costanti di tempo introdotte dai circuiti elettronici d'interfaccia e, nei modelli digitali, dal tempo di quantizzazione. Quest'ultimo, lo ricordiamo, è il tempo che deve spendere il convertitore A/D per trasformare una grandezza discreta in un dato. Qualunque sia il tipo di trasduttore, o le sue prestazioni, dobbiamo tenere presente che è inserito in un sistema complesso destinato, nel suo insieme, a regolare una grandezza fisica (velocità, accelerazione, posizione di una massa mobile, ecc.). Senza troppo dilungarmi sulla complicata teoria dei sistemi di controllo è però necessario, a questo punto, fornire almeno un'idea del contesto dove il trasduttore opera. La figura 4 mostra un modello a blocchi della tipica struttura di un sistema di controllo in cui una grandezza fisica, nello schema indicata dalla lettera C, è controllata da un insieme elettronico-meccanico complesso.

La grandezza di riferimento R, generalmente una tensione o una corrente, posta all'ingresso della catena di controllo, presiede alla regolazione manuale della grandezza in uscita. In altri termini, la manipolazione di R consente di impostare, su valori scelti e conosciuti, la grandezza fisica C. Si noti come il segnale d'uscita fornito dal trasduttore si sottrae, all'interno del nodo sommatore, alla grandezza di riferimento. Il risultato di questa somma algebrica è posto all'ingresso del blocco amplificatore, quest'ultimo necessario in quanto i segnali in gioco sono di ampiezza e potenza contenuti. All'uscita dell'amplificatore è posto il blocco denominato attuatore che ha il compito, affatto trascurabile, di convertire i segnali elettrici in un movimento meccanico. Il blocco denominato sistema, rappresentante la parte fisica vera e propria, è composta dall'insieme di organi meccanici, cinematismi, ecc., che producono direttamente la variabile controllata C all'uscita di tutta la catena. Una minima parte di quest'ultima grandezza è posta all'ingresso del trasduttore che, a mezzo della sua caratteristica di trasferimento, genera un segnale elettrico (T nello schema) proporzionale all'entità di C. Per fare un esempio concreto,

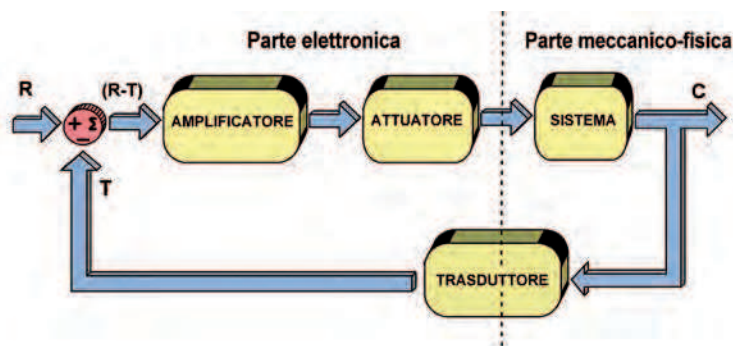



Figura 4
Schema a blocchi di un sistema di controllo ad anello chiuso

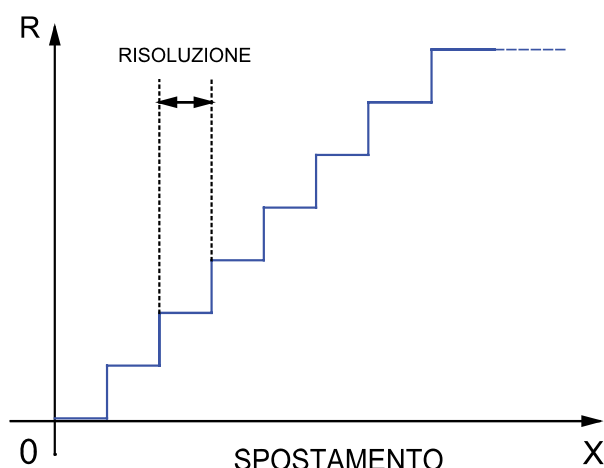



Figura 5
Tipica caratteristica di un potenziometro a filo

sfiorando appena la complessa teoria dei controlli automatici, possiamo immaginare, in un'imbarcazione, il sistema di comando del timone a mezzo della barra. In questo caso R sarà una tensione proporzionale all'angolo di barra che, scelto in funzione della rotta, deve rimanere costante il più possibile durante la navigazione. Un amplificatore si occuperà di convertire queste tensioni di controllo in un movimento del timone a mezzo, ad esempio, di un motore elettrico, quest'ultimo simboleggia il nostro attuatore. Il movimento del timone è ricondotto al trasduttore di posizione che ha il compito, importantissimo, di "tradurre" questo spostamento in segnali elettrici, a loro volta riportati all'ingresso invertente del nodo sommatore. In tali condizioni, se la posizione del timone è esattamente quella impostata a mezzo del segnale R, semplicemente non accadrà nulla poiché la grandezza controllata, ovvero lo spostamento del timone, rientra nella rotta da noi scelta. Supponiamo però che un fattore perturbativo, ad esempio la corrente marina, ad un certo punto introduca un errore nella posizione del timone. Senza scomodare formuloni matematici possiamo affermare che il sistema di controllo reagirà automaticamente ripristinando l'esatta posizione del timone, non sarà necessario, quindi, riposizionare manualmente la barra (ritoccando la grandezza di riferimento R). Il timone, in pratica, sarà continuamente collocato sulla giusta rotta da piccoli e ripetuti azionamenti del motore elettrico. Se il sistema non avesse avuto a disposizione il segnale fornito dal trasduttore, nella figura 4 definito T, per ripristinare la rotta giusta sarebbero stati necessari ritocchi continui della grandezza R (con conseguente iperattività del "nocchiero"). Quanto visto può anche definirsi un sistema di controllo ad anello chiuso, ovvero controreazionato. Quest'ultimo termine, sicuramente già noto alla maggior parte dei lettori, indica sempre una sommatoria di una parte del segnale d'uscita con quello in ingresso. La "chiusura" dell'anello di regolazione è affidata, per l'appunto, al trasduttore. Utilizzare catene di controllo controreazionate consente di correggere automaticamente eventuali errori dovuti a imprecisioni, disturbi esterni, ecc. che, in condizioni di anello aperto, porterebbero la grandezza controllata fuori dai valori di regolazione impostati manualmente. Dopo questa necessaria introduzione esaminiamo le principali categorie di trasduttori di posizione, cercando di capirne il funzionamento e le loro caratteristiche.

Potenziometri

Cominciamo la nostra trattazione con il componente che, almeno nelle sue forme più semplici, è conosciuto da tutti. Come accennato, almeno teoricamente, un potenziometro può restituire per ogni infinitesimo grado di rotazione del suo perno un altrettanto infinitesima variazione del suo valore di resistenza. Si dirà, quindi, che il potenziometro è in grado di fornirci una risposta non quantizzata, ovvero continua, della sua grandezza in uscita (un valore resistivo). In realtà, come al solito, le cose non vanno poi così bene. Intanto il potenziometro ha un organo importantissimo destinato a stabilire il contatto mobile, il cursore per l'appunto, che ha dimensioni fisiche non trascurabili rispetto alla lunghezza totale del percorso resistivo. Questa particolarità impedisce, com'è facile immaginare, la restituzione di valori resistivi fra loro molto vicini. Inoltre, la maggior parte dei potenziometri ha come elemento resistivo una striscia di grafite (di forma lineare o angolare) che, durante il funzionamento, a causa del contatto strisciante, tende ad usurarsi riducendo il suo spessore. Ne consegue che la stessa posizione, a distanza di tempo, potrebbe non restituire più lo stesso valore di resistenza. Inoltre, le piccole scabrosità formate per effetto dell'attrito, si traducono, all'atto pratico, in un forte rumore che disturba i dispositivi elettronici di lettura. Il problema dell'usura diviene poi un fattore molto critico se il potenziometro deve operare,

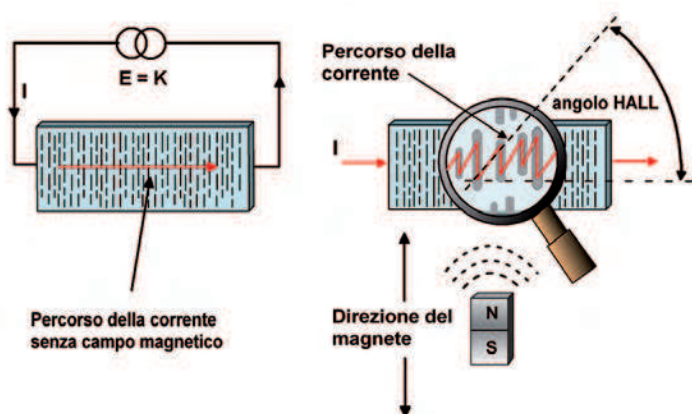



Figura 6
Principio di funzionamento di un potenziometro magnetico

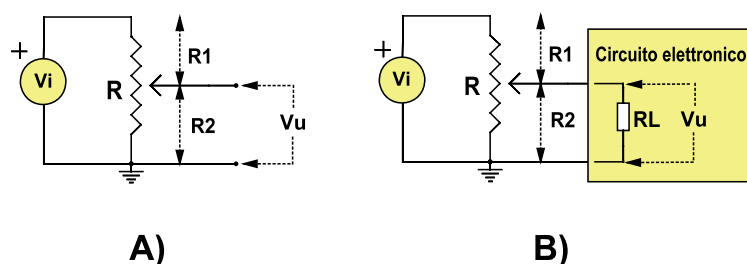



Figura 7 A e B
Potenziometro con funzionamento ideale (a) e potenziometro caricato dal circuito (b)

Si noti come la tensione d'uscita, proporzionale allo spostamento del cursore, dipende dal rapporto fra il valore delle due resistenze create dal cursore stesso, nel disegno R1 ed R2. Nelle condizioni ideali (figura 7a), ovvero senza carico inserito a valle del cursore, vale la relazione:

$$Vu = Vi \times (R2 / R)$$

In cui R, rappresentando il valore del potenziometro, è costante mentre R2 varia in funzione dello spostamento. Nella figura 7b, invece, il cursore del potenziometro è caricato dal resistore RL che simula l'impedenza d'ingresso del circuito preposto all'elaborazione della tensione fornita dal trasduttore. Si noti come RL viene a trovarsi in parallelo alla resistenza R2, modificandone il valore indipendentemente dalla posizione del cursore. In queste condizioni la relazione diviene:

$$Vu = Vi \times Rp / (R1 + Rp)$$

come nel caso di macchine utensili, a velocità angolari (o lineari) molto elevate compiendo, durante i cicli operativi, anche migliaia di movimenti rotatori completi. Per ovviare, almeno in parte, a tali limitazioni la tecnologia ci viene incontro in due modi. Intanto sono reperibili potenziometri che, al posto della classica pista di grafite, hanno un avvolgimento filare, composto generalmente di tungsteno di appropriata sezione. L'uso di questo metallo, noto anche per la sua durezza, ovvia in buona parte ai problemi legati all'usura di movimento. Per contro, il fatto che il cursore adesso si muove sul lato di un avvolgimento, implica una variazione resistiva non continua bensì a gradini che, colpo di scena, sembra di tipo digitale (vedi figura 5).
La scelta di un potenziometro, sia esso a filo o a grafite, come elemento trasduttore rappresenta un buon compromesso fra prezzo, di solito mai troppo elevato, e prestazioni. La piccola massa del cursore e la tecnologia costruttiva consentono elevatissime risposte in frequenza, per contro, i potenziometri tradizionali, non offrono elevate garanzie di durata ed anche la loro linearità è soggetta a peggioramento nel tempo. Inoltre, i potenziometri a filo ed a grafite, necessitano di circuiti elettronici d'interfaccia ad alta impedenza d'ingresso (generalmente amplificatori operazionali). Vediamo di capirne il perché osservando la figura 7ab, che mostra una tipica configurazione d'impiego di questi componenti.

In cui:

$$Rp = (R2 \times RL) / (R2 + RL)$$

Come possiamo facilmente dedurre, la proporzionalità fra movimento del cursore e tensione d'uscita non è più legata semplicemente al valore di R2. Nel calcolo, infatti, entra a far parte anche RL, sotto forma di parallelo con R2, e tale influenza può essere ignorata soltanto quando il potenziometro ha il cursore fisso ad uno dei due fine corsa, ovvero quando R2 = R e R2 = 0. Salvo queste due eccezioni, in tutti gli altri casi la presenza di RL introdurrà una accentuata mancanza di linearità che sarà massima per valori in cui R1 = R2 (cursore a metà corsa). Per evitare questo inconveniente, se non è possibile elevare l'impedenza d'ingresso del circuito, è giocoforza scegliere un potenziometro di valore almeno venti volte inferiore ad RL.

LVDT

La sigla LVDT non indica una particolare sezione dei Servizi Segreti ma, più tecnicamente, è l'acronimo di Linear Voltage Differential Transformer. Nell'italico idioma: trasformatore lineare differenziale di tensione. Questi dispositivi, presenti nel mercato da molti anni, sono sostanzialmente dei trasformatori provvisti di un traferro mobile e di 3 avvolgimenti (vedi figura 8a).
L'avvolgimento centrale, alimentato da una sor-

Figura 8
Principio di funzionamento di un LVDT (a) e sua curva caratteristica (b)

Figura 9
Esempi commerciali di trasduttori LVDT

gente in CA a frequenza compresa fra 50 e 250 KHz, varia il suo accoppiamento induttivo con i due secondari in funzione dello spostamento del traferro. Lo spostamento assiale di quest'ultimo fornirà, quindi, un segnale in CA con un andamento simile a quello mostrato in figura 8b. Il particolare andamento della funzione, nei confronti dello spostamento X, dipende dal fatto che i secondari pur essendo collegati in serie sono in opposizione di fase. Si noti come il segnale risultante, nella posizione di simmetria del nucleo mobile, è praticamente zero. Generalmente gli LVDT, commercializzati con forme simili a quelle mostrate in figura 9, vengono usati per misurare spostamenti lineari anche di diverse decine di centimetri.
Esiste anche una versione particolare di LVDT, adatta per spostamenti angolari, leggermente più complessa. Lo schema costruttivo di questo dispositivo è mostrato in figura 10. Come si noterà, anche in questo caso vi è un nucleo mobile, definito rotore e sprovvisto di avvolgimenti, e ben 4 nuclei fissi che fanno corpo unico con lo statore provvisti, questi ultimi, di 2 avvolgimenti cadauno.

Si noti come ogni singola espansione polare sia provvista di doppio avvolgimento: uno relativo alla tensione di alimentazione e l'altro alla tensione d'uscita. Tutti gli avvolgimenti, poi, sono posti in serie fra loro. Quando viene compiuto un movimento angolare la tensione d'uscita subisce variazioni di ampiezza e fase, quest'ultima grandezza definisce anche il senso di rotazione. La tecnologia permette di ottenere, nei modelli di punta, anche risoluzioni di 0,05 V per ogni grado di rotazione. Per contro, la parte elettronica incaricata di trasformare il segnale CA in una tensione continua proporzionale all'angolo di rotazione, è abbastanza complessa. Tale circostanza è dovuta a tre fattori: l'uso di tensioni in CA a frequenza relativamente elevata, la non perfetta linearità dei trasformatori impiegati ed i flussi dispersi che causano errori di off-set e di lettura. Inoltre, l'ampiezza delle tensioni rilevate sui secondari non permette, di per se, di conoscere la direzione di spostamento del nucleo. Per ottenere questa informazione, sempre necessaria, occorre effettuare un confronto di fase fra le tensioni in uscita e quella fornita al primario. Tra l'altro, avere a che fare con tensioni CA implica l'uso di demodulatori necessari per il processo di conversione del segnale in CC. Questi circuiti possiedono costanti di tempo relativamente alte che, di fatto, riducono apprezzabilmente la banda passante di questa tipologia di trasduttori.

Trasduttori magnetoelastici

I trasduttori che sfruttano le proprietà magnetoelastiche di alcuni materiali ferromagnetici, ad esempio il nichel, sono di fatto "l'ultimo grido" nel variegato mondo dei sensori di posizione. La teoria di funzionamento di questi dispositivi è leggermente complessa e si basa su un effetto, noto da tempo, che fa modificare la struttura fisica di alcuni materiali quando sono interessati da un campo magnetico. Questi piccolissimi cambiamenti

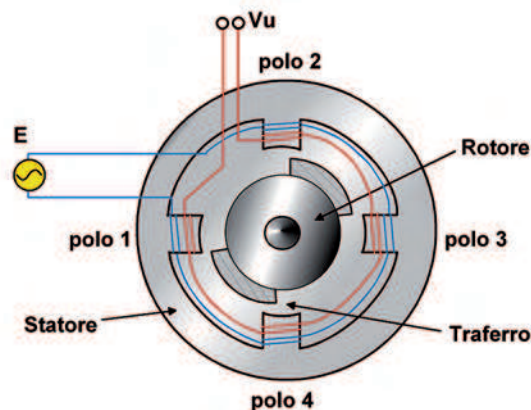



Figura 10
Principio di funzionamento di un LVDT angolare

dimensionali, naturalmente, sono del tutto reversibili e cioè il materiale ritorna nelle sue dimensioni originali non appena il campo magnetico cessa. In figura 11 è rappresentato, molto schematicamente, uno di questi trasduttori.
Il cuore del dispositivo è il sottile film plastico ricoperto da un metallo con proprietà magnetoelastiche, generalmente è usato il nichel, intorno al quale vi è un avvolgimento. Questo componente ha la doppia funzione di trasmettere un impulso a frequenza sonica, sempre inferiore a 20 KHz, e poi, come vedremo, di riceverne l'"eco" comportandosi come un vero e proprio pick-up. Insomma quasi un RADAR in un tubo di pochi centimetri! Ora cerchiamo di capire meglio cosa avviene. Quando il pick-up emette un impulso, generalmente con durata non superiore a 2-3 μs, esso è trasmesso alla guida d'onda (in realtà un semplice conduttore) grazie all'accoppiamento meccanico fra quest'ultima ed il pick-up. Da questo istante, che chiameremo START, l'impulso viaggia sulla guida d'onda alla velocità del suono (circa 3000 m/s) e, se non incontrasse ostacoli, si smorzerebbe sulla terminazione assorbente. Eppure un ostacolo c'è: il campo prodotto dal magnete permanente montato coassiale alla guida d'onda. Quest'ultimo, come si evince dal disegno, è libero di muoversi e solidale con l'organo di cui dobbiamo conoscere la posizione. Quando il campo magnetico è raggiunto dall'impulso, l'interazione che ne consegue "rispedisce" indietro una parte dell'energia trasmessa sotto forma di vibrazione acustica. Tale onda di ritorno è rilevata dal pick-up, ora nella sua condizione passiva, nell'istante che chiameremo STOP. A questo punto è sufficiente misurare la differenza di tempo fra l'istante START e STOP per avere, dopo opportuna elaborazione, la posizione del magnete permanente. I trasduttori magnetoelastici, in inglese magnetostrictive transductor, sono usati ogni qual volta occorre una grande risoluzione (si arriva anche al μm) ed alta affidabilità di misura. Alcuni modelli, particolarmente sensibili, vengono impiegati anche nel campo sismografico e per individuare piccole inclinazioni, ovvero cedimenti, di infrastrutture o edifici. La parte elettronica che presiede l'elaborazione del segnale fornito dal pick-up, com'è facile intuire, è abbastanza complessa. Spesso, per ragioni di ottimizzazione degli spazi, si preferisce installare il circuito all'esterno del trasduttore vero e proprio.

Esistono poi particolari potenziometri, detti magnetici, che basano il loro funzionamento sul principio di Thomson e che sono privi di contatti striscianti. In questi modelli il percorso resistivo, composto da una sottile piastrina di materiale semiconduttore, varia la propria resistenza se interessata ad un campo magnetico. Quando quest'ultimo, fornito ad esempio da un magnete permanente, investe la piastrina il percorso della corrente all'interno del semiconduttore si modifica secondo un certo angolo proporzionale all'intensità del flusso magnetico stesso (vedi figura 6).
Un percorso più lungo della corrente si tradurrà, in termini pratici, in una maggiore resistenza dell'elemento semiconduttore. Questi potenziometri, pur risolvendo una buona parte dei problemi che affliggono i loro "fratelli" tradizionali, presentano una caratteristica non esattamente lineare che può essere corretta solo con opportuni circuiti.

Figura 11
Raffigurazione del funzionamento di un trasduttore magnetoelastico

SYNCRO e resolver

I resolvers ed i SYNCRO sono dei trasduttori angolari aventi, fra loro, principio di funzionamento molto simile. I resolvers, in particolare, oltre ad eseguire la normale conversione dei gradi in grandezza elettrica, possono effettuare calcoli analogici trigonometrici (seno, coseno, ecc.) e anche trasformare coordinate polari in cartesiane. Osservando lo schema in figura 12a, si può notare che sono essenzialmente composti da due avvolgimenti esterni posti in quadratura spaziale, ovvero con assi elettromagnetici disposti a 90°, e di un avvolgimento interno rotante.
L'avvolgimento interno, solidale con

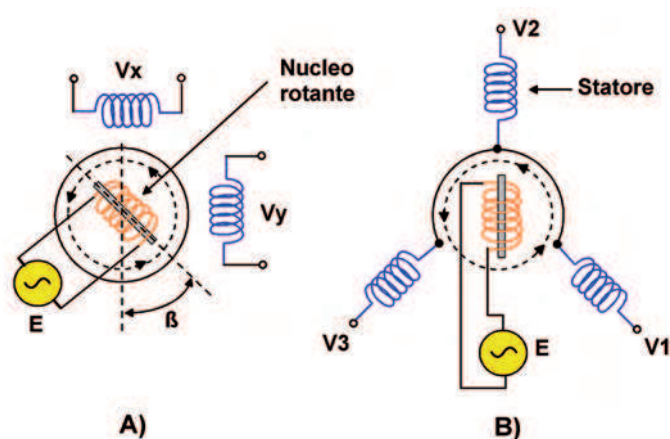



Figura 12
Schema di funzionamento di un resolver (a) e di un SYNCRO (b)

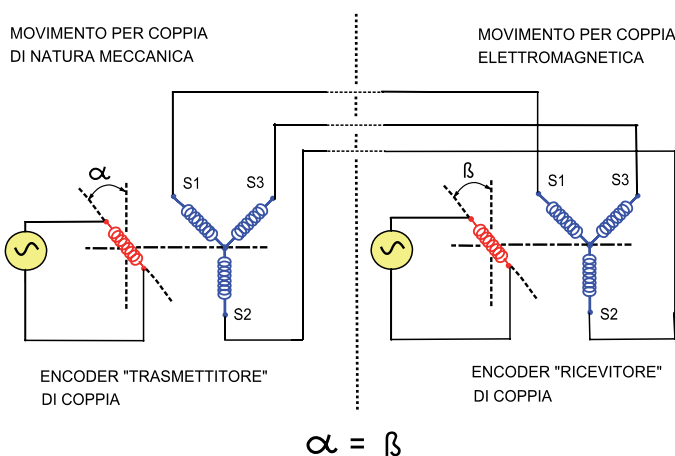



Figura 13
Funzionamento di due syncro accoppiati

l'asse di rotazione, è chiamato rotore mentre, quelli esterni, fissi, statori. Nella figura 12b è schematizzato un SYNCRO che, rispetto ad un resolver, ha un avvolgimento in più sullo statore ed i tre avvolgimenti sono posti, stavolta, a 120° (radiali rispetto al centro di rotazione). Generalmente la DDP di eccitazione (di solito la tensione di rete) è applicata al rotore a mezzo anelli rotanti o, nei modelli più perfezionati, induttivamente. Questa impostazione costruttiva non è tassativa e vengono prodotti trasduttori in cui la DDP è applicata agli statori mentre i rotori, per incrementare la risoluzione angolare, possono anche avere anche due avvolgimenti (posti in quadratura elettromagnetica). Nel loro funzionamento syncro e resolvers si comportano come dei trasformatori rotanti, in cui le tensioni d'uscita sono in funzione della tensione d'eccitazione e della posizione angolare del rotore. Con riferimento allo schema di figura 12a, è possibile scrivere le seguenti relazioni:

$$Vx = KE \operatorname{sen} \beta \text{ e } Vy = KE \cos \beta$$

In cui K è un coefficiente di proporzionalità relativo al tipo di trasduttore e ß l'angolo di rotazione del rotore rispetto ad un immaginario asse verticale. Combinando fra loro, a mezzo appositi circuiti, le uscite Vx e Vy è possibile ricavare funzioni goniometriche complesse. Queste ultime caratteristiche rendono questa categoria di trasduttori molto adatta a fungere da elemento di retroazione nei sistemi di controllo e posizionamento per motori in CA e CC. La particolare tecnologia costruttiva dei resolvers, tra l'altro, consente di usare questi dispositivi accoppiati, con il vantaggio di escludere, per piccole potenze in gioco, l'elemento attuatore (vedi figura 4). Infatti è possibile alimentare gli avvolgimenti di un resolver, che possiamo definire "ricevente" con la DDP presente ai capi degli stessi avvolgimenti di un resolver "trasmittente". Con la configurazione schematizzata in figura 13 qualunque movimento angolare del "trasmittente" sarà riprodotto fedelmente dal "ricevente".

Syncro e resolver, a causa delle loro particolarità costruttive, sono abbastanza insensibili a disturbi di natura elettrica. Per questo motivo sono molto impiegati nell'avionica, nella missilistica e ovunque sia necessario controllare cinematismi asserviti elettricamente con ottima performance e precisione. A dimostrazione di quanto sia importante questo tipo di trasduttore vi è il fatto che, alcune aziende, commercializzano chips dedicati alla conversione numerica dell'informazione di posizione fornita dagli avvolgimenti d'uscita.

Conclusioni

Anche se gli argomenti esposti possono sembrare confinati nella teoria, in realtà, capire il funzionamento di questi importantissimi componenti, prospettandoli nello studio dei sistemi di controllo, aiuta ad organizzare le nostre informazioni in modo più organico e produttivo. nella prossima puntata parleremo dei commutatori codificati e degli encoder ottici, mentre, nella puntata ancora successiva sarà presentata un'applicazione pratica che fa uso di un encoder di tipo elettromeccanico.

Imparare a progettare

Per motivi connessi alle impedenze in gioco in un circuito, molto spesso è necessario ricorrere agli inseguitori di tensione. Qui si illustrano le modalità di progetto dell'emitter follower e del source follower

Generalità

L'inseguitore di tensione – anche definito voltage follower – è un particolare amplificatore che presenta guadagno unitario, segnale di uscita in fase col segnale di ingresso, alta impedenza di ingresso e bassa impedenza di uscita. Ogniqualvolta si desideri un adattamento di impedenza si farà pertanto ricorso all'inseguitore di tensione che può essere realizzato secondo varie configurazioni circuitali, utilizzando BJT (*emitter follower*) o FET e Mosfet (*source follower*) o amplificatori operazionali (genericamente definiti voltage follower).

Voltage follower con BJT (emitter follower)

Per l'inseguitore di tensione che utilizzi quale componente attivo il BJT – figura 1 – si hanno le seguenti formule di progetto:

$R_e = (0,5 \cdot V_a) / I_c$ [1]

$I_b = I_c / h_{FE}$ [2]

$I_k = (10 \div 30) \cdot I_b$ [3]

$R_1 = [(0,5 \cdot V_a) - 0,7)] / I_k$ [4]

$R_2 = [0,7 + (0,5 \cdot V_a)] / (I_k - I_b)$ [5]

$Z'_i = h_{ie} + h_{fe} \cdot (R_e // R_L)$ [6]

$Z_i = Z'_i // R_B \quad con \quad R_B = R_1 // R_2$ [7]

$Z'_o = [(h_{ie} / h_{fe}) // R_e]$ [8]

$C_1 = 0,16 / (f_L \cdot Z_i)$ [9]

$C_2 = 1,6 / [f_L \cdot (R_e + R_L)]$ [10]

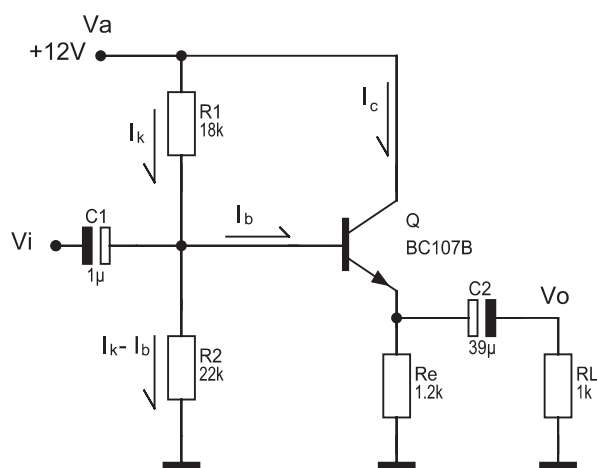
$Z_o = Z'_o // R_L$ [11]

$G = 1 - (h_{ie} / Z'_i)$ [12]

Figura 1 a
Voltage follower con BJT (emitter follower)

Esempio 1

Si dimensioni l'emitter follower della figura 1 per f_L = 20 Hz; si calcoli quindi l'impedenza di uscita e il guadagno. Sono dati: 1_{ipp} = 1 V e 1_L = 1 kΩ.

Soluzione

Si può porre, per esempio V_a = 12 V e I_c = 5 mA. A questa corrente di collettore corrispondono, per il BJT BC107B qui utilizzato (v. diagrammi *A* e *B*, primo articolo), i parametri:

$h_{fe} = 360$

$h_{FE} = 320$

$h_{ie} = (0,026 \times 360) / (5 \times 10^{-3}) = 1872\Omega$

Dalla [1] si ricava quindi la resistenza di emettitore:

$R_e = (0,5 \times 12) / (5 \times 10^{-3}) = 1,2\ k\Omega$

Tramite la [2] si calcola la corrente di base del BJT:

$I_b = (5 \times 10^{-3}) / 320 = 15,6\ \mu A$

Ponendo, dalla [3], $I_k = 20\ I_b = 20 \times 15,6 \times 10^{-6} = 312\ \mu A$, si ha:

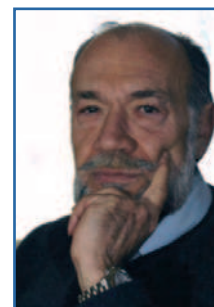
$I_k - I_b = (312 - 15,6) \times 10^{-6} = 296,4\ \mu A$

Con le espressioni [4] e [5] si calcola il valore da attribuire alle resistenze del partitore di ingresso R_1, R_2. Si ricava:

$R_1 = (6 - 0,7)/(312 \times 10^{-6}) = 16,98\ k\Omega \quad \Rightarrow \quad R_1 = 18\ k\Omega$

$R_2 = (0,7+6)/(296,4 \times 10^{-6}) = 22,6\ k\Omega \quad \Rightarrow \quad R_2 = 22\ k\Omega$

Essendo $R_e//R_L$ = 546 Ω, con l'espressione [6] si cal-

L'inseguitore di tensione con BJT, Fet e Mosfet

di Nico Grilloni

cola l'impedenza di ingresso Z'_i a valle del partitore. Si ha:

$Z'_i = 1872 + (360 \times 546) = 198{,}4\ k\Omega$

Essendo $R_B = R_1 // R_2 = 9{,}9\ k\Omega$, tramite la [7] si calcola l'impedenza di ingresso dello stadio:

$Z_i = (198400 \times 9900) / (198400 + 9900) = 9431\ \Omega$

Con l'espressione [9] si calcola quindi la capacità del condensatore di ingresso:

$C_1 = 0{,}16 / (20 \times 9431) = 0{,}848\ \mu F \quad \Rightarrow \quad C_1 = 1\ \mu F$

Poiché:

$h_{ie} / h_{fe} = 1872 / 360 = 5{,}2\ \Omega$

con la [8] si calcola l'impedenza di uscita a monte del carico R_L. Si ricava:

$Z'_o = 5{,}2 // 1200 = 5{,}177\ \Omega$

Si noti che essendo $R_L >> Z'_o$, l'impedenza di uscita Z_o coincide in pratica, per la [11], con la Z'_o che, a sua volta, è pressoché coincidente con il rapporto h_{ie} / h_{fe}. Tramite la [10], si determina quindi il valore della capacità C_2:

$C_2 = 1{,}6/[20 \times (1200+1000)] = 36{,}4\ \mu F \quad \Rightarrow \quad C_2 = 39\ \mu F$

Infine, con la [12], si calcola il guadagno dello stadio:

$G = 1 - (1872 / 198400) = 0{,}99$

La figura 1 riporta l'emitter follower su dimensionato, mentre la figura 2 riporta le forme d'onda di ingresso e di uscita. La V_i e la V_o sono in fase e l'ampiezza da picco a picco di quest'ultima, pari a 987,5 mV, è pertanto di poco inferiore all'ampiezza della tensione di ingresso (1000 mV). La posizione reciproca dei marker verticali *a* e *b* indica la frequenza della tensione V_i (e quindi della V_o).

La figura 3 infine, riporta la risposta in frequenza dello stadio. A –3 dB, la frequenza di taglio inferiore f_L, indicata dal marker verticale *a*, è pari a 18,15 Hz. Si legge, infatti: *Xa* = 18,15 Hz.

Determinazione della tensione di alimentazione

Si può assegnare alla resistenza R_e un qualsiasi valore compreso fra qualche centinaio e qualche migliaio di ohm. Quindi, noto il carico R_L, può porsi:

$I_c \geq V_{opp} / (R_e // R_L)$ [13]

Si determina quindi la tensione di alimentazione con l'espressione:

$V_a = 2 \cdot R_e \cdot I_c$ [14]

Nell'esempio 1 si poteva quindi – essendo il valore da picco a picco della tensione di uscita pari a 1 V – porre, per esempio, $R_e = 1{,}5\ k\Omega$ e quindi, calcolato il parallelo $R_e // R_L = 600\ \Omega$ (per $R_L = 1\ k\Omega$), determinare la corrente di collettore con l'espressione [13]:

$I_c = 1 / 600 = 1{,}67\ mA$

Quindi, tramite la [14], per la tensione di alimentazione si ha il valore:

$V_a = 2 \times 1500 \times 1{,}67 \times 10^{-3} = 5\ V$

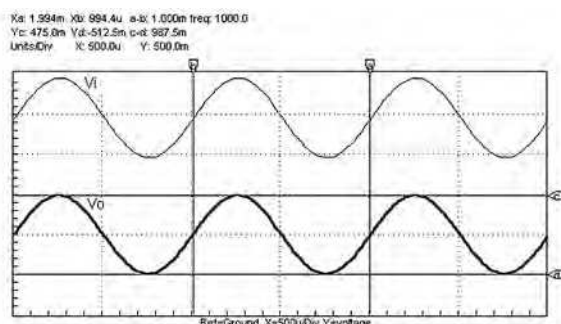



Figura 2
Risposta in ampiezza dell'emitter follower di cui alla figura 1

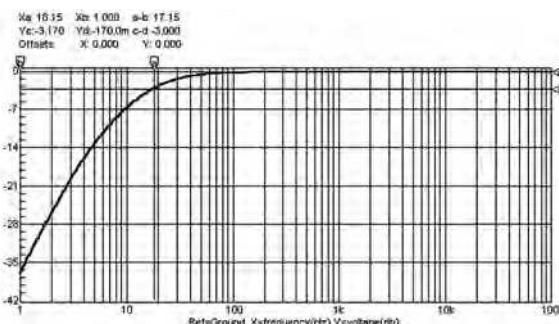



Figura 3
Risposta in frequenza dell'emitter follower di cui alal figura 1

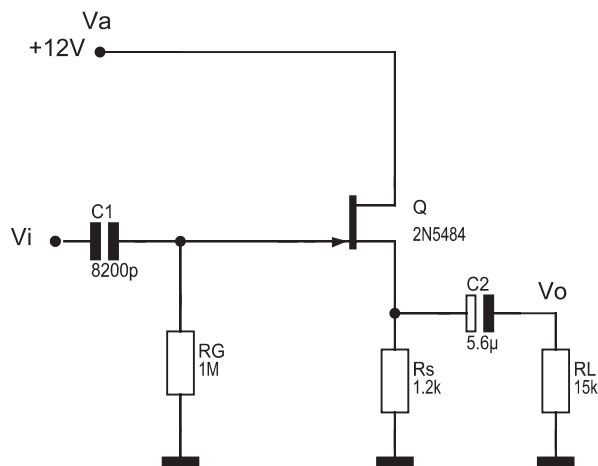



Figura 4
Lo schema del source follower

Calcolata poi la corrente di base tramite la [2], e attribuito alla corrente I_k del partitore un qualsiasi valore compreso fra dieci e trenta volte la I_b, con le espressioni [4] e [5] si procederà al calcolo delle resistenze R_1 e R_2.

Voltage follower con FET (source follower)

Le formule di progetto per il voltage follower a FET, detto anche inseguitore di source (source follower), sono:

$$V_{GS} = V_{GS(off)} \cdot \left(1 - \sqrt{\frac{I_D}{I_{DSS}}}\right) \quad [15]$$

$$R_S = V_{GS} / I_D \quad [16]$$

$$C_1 = 0,16 / (f_L \cdot R_G) \quad [17]$$

$$C_2 = 1,6 / [f_L \cdot (R_S + R_L)] \quad [18]$$

$$Z_i = R_G \quad [19]$$

$$Z_o = R_S / [1 + (g_{fs} \cdot R_S)] \quad [20]$$

$$G = [g_{fs} \cdot (R_S // R_L)] / [1 + g_{fs} \cdot (R_S // R_L)] \quad [21]$$

$$V_a = V_{DS} + R_s I_D \quad [22]$$

Esempio 2

Si dimensioni il source follower della figura 4 per f_L = 20 Hz. Sono dati: V_{ipp} = 0,4 V e R_L= 15 kΩ.

Soluzione

Facendo uso del FET a canale *n* 2N5484, imponendo una corrente I_D di drain pari a 1 mA, si hanno i parametri utili:

$g_{fs} = 3\ mmho$
$I_{DSS} = 3\ mA$
$V_{GS(off)} = 3\ V$

Tramite la [15] si calcola la d.d.p. V_{GS}:

$$V_{GS} = 3 \times \left(1 - \sqrt{\frac{1}{3}}\right) = 1,26\ V$$

Quindi, dalla [16] si ricava il valore della resistenza di source:

$R_S = 1,26 / (1 \times 10) = 1,26\ k\Omega \quad \Rightarrow \quad R_S = 1,2\ k\Omega$

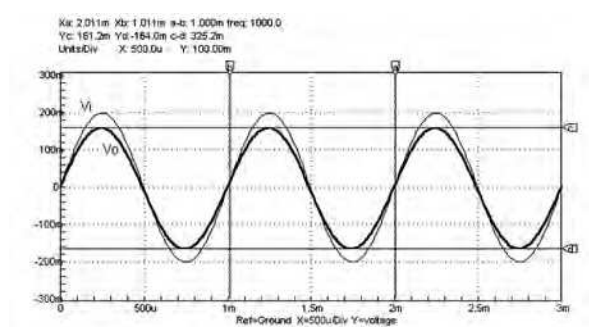



Figura 5
Risposta in ampiezza del source follower di cui alla figura 4

Dalle espressioni [17] e [18] si ricavano i valori delle due capacità C_1 e C_2:

$C_1 = 0,16/(20 \times 1000000) = 0,008\ \mu F \quad \Rightarrow \quad C_1 = 8200\ pF$
$C_2 = 1,6/[20 \times (1200+15000)] = 4,93\ \mu F \quad \Rightarrow \quad C_2 = 5,6\ \mu F$

L'impedenza di ingresso – espressione [19] – è pari a 1 MΩ, mentre l'impedenza di uscita, per la [20] vale:

$Z_o = 1200 / [1 + (3 \times 10^{-3} \times 1200)] = 260,8\ \Omega$

Il guadagno G, per la [21], essendo $R_S // R_L$=1111 Ω, ha il valore:

$G = (3 \times 10^{-3} \times 1111) / [1 + (3 \times 10^{-3} \times 1111)] = 0,77$

Le figure 5 e 6 riportano, rispettivamente, la risposta in ampiezza e la risposta in frequenza. Nella prima si vede che il segnale di uscita vale, da picco a picco, 325,2 mV. Essendo V_{ipp} = 400 mV si ha, per G, un valore di poco diverso dal valore teorico calcolato. Si ricava infatti:

$G = 325,2 / 400 = 0,81$

Si constata così che il source follower ha una perdita di segnale maggiore dell'emitter follower. Con quest'ultimo infatti è sufficientemente facile ottenere un segnale di uscita di ampiezza assai prossima all'ampiezza del segnale di ingresso. La maggior perdita del source follower è però compensata da una impedenza di ingresso notevolmente più elevata.

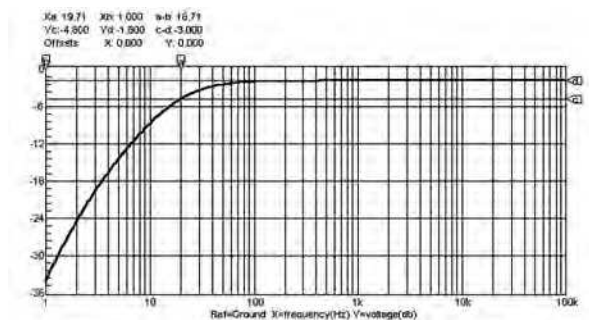



Figura 6
Risposta in frequenza del source follower di cui alla figura 4

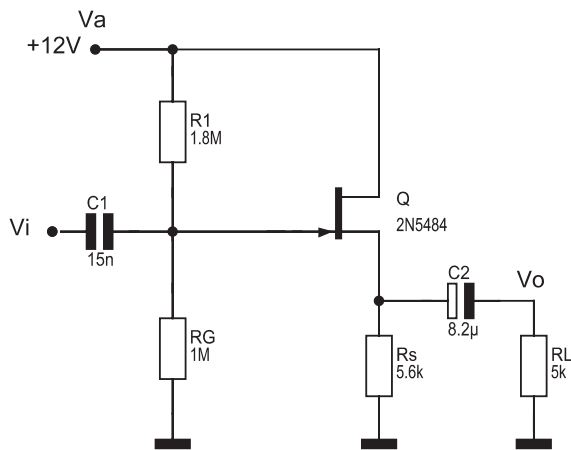



Figura 7
Source follower con polarizzazione a partitore

La curva di risposta in frequenza – figura 6 – evidenzia come a - 3 dB la frequenza f_L sia entro i limiti indicati dalle specifiche di dimensionamento. È infatti: $f_L = 19{,}71$ Hz. Si legge, infatti: $Xa = 19{,}71$ Hz.

Source follower con polarizzazione a partitore

Come il preamplificatore a source comune, anche il source follower può polarizzarsi tramite il partitore R_1, R_2 così come illustra la figura 7.
Per il dimensionamento si fissano la tensione di alimentazione e la corrente di drain. Si impone quindi una c. d. t. sulla resistenza di source R_S pari a metà della tensione di alimentazione e si calcola la resistenza R_1 come già si è proceduto nel caso del premplificatore a FET con identica polarizzazione.

Esempio 3

Per il circuito della figura 7 si è utilizzato il FET a canale *n* 2N5484. Alimentato il circuito a 12 V e fissata una corrente di drain di 1 mA, si è imposta una c. d. t. sulla R_S pari a metà della V_a. Per R_S si ricava quindi:
$R_S = (0{,}5 \cdot V_a)/I_D = 6/(1\times10^{-3}) = 6\ k\Omega \quad \Rightarrow \quad R_S = 5{,}6\ k\Omega$

Con il valore normalizzato di 5600 Ω si ha, sulla stessa R_S, una c. d. t. pari a:
$V_{RS} = 5600 \times 1 \times 10^{-3} = 5{,}6\ V$

Per I_D = 1 mA, il FET 2N5484 presenta per i parametri utili, i valori:
$g_{fs} = 3\ mmho$
$I_{DSS} = 3\ mA$
$V_{GS(off)} = 3\ V$

Per la tensione V_{GS}, in base alla [15], si ricava allora:

$$V_{GS} = 3 \times \left(1-\sqrt{\frac{1}{3}}\right) = 1{,}26\ V$$

Pertanto, per la d. d. p. V_{GM} fra gate e massa si ricava (v. FE n. 264):
$V_{GM} = 5{,}6 - 1{,}26 = 4{,}34\ V$

Attribuito alla R_2 il valore, per esempio, di 1 MΩ, si ha per R_1 (v. FE n. 264):
$R_1 = [1000000\times(12-4{,}34)]/4{,}34 = 1{,}76\ M\Omega \Rightarrow R_1 = 1{,}8\ M\Omega$

Si calcola quindi il parallelo $R_1//R_2$. Si ha $R_1//R_2$ = =643 kΩ. Essendo questa l'impedenza di ingresso dello stadio, per la capacità C_1, dalla [17] si ricava:
$C_1 = 0{,}16/(20\times643000) = 0{,}0124\ \mu F \Rightarrow C_1 = 0{,}015\ \mu F$

Dall'espressione [4.18] si ricava poi la capacità di uscita C_2:
$C_2 = 1{,}6/[20\times(5600+5000)] = 7{,}54\ \mu F \Rightarrow C_2 = 8{,}2\ \mu F$

La figura 7 riporta il source follower su dimensionato, mentre le figure 8 e 9 riportano, rispettivamente, la risposta in ampiezza e la risposta in frequenza.

Nella prima si constata come l'ampiezza da picco a picco della tensione di uscita, indicata dalla differenza (*c* - *d*) nella posizione degli omonimi marker, sia pari a 365 mV; per V_{ipp} = 400 mV, il guadagno dello stadio, essendo $G = V_{opp}/V_{ipp}$, vale allora:
$G = 365 / 400 = 0{,}91$

Nella seconda si vede come a -3 dB, differenza (*c* - *d*) nella posizione degli omonimi marker orizzontali, la frequenza di taglio f_L – indicata dal marker verticale *a* all'intersezione con la curva di risposta – sia pari a 17,78 Hz.

Figura 8
Risposta in ampiezza del source follower di cui alla figura 7

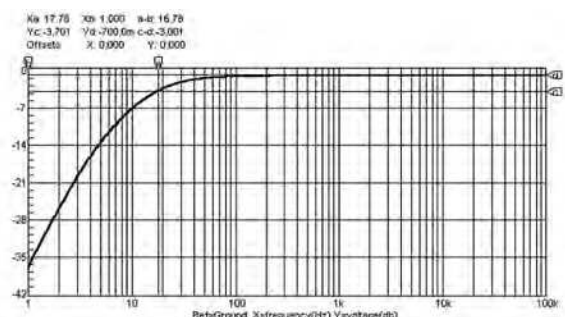



Figura 9
Risposta in frequenza del source follower di cui alla figura 7

Source follower con Mosfet

La figura 10 riporta, quindi, un voltage follower realizzato con Mosfet. Per il dimensionamento, dopo aver imposto che sulla resistenza di source ci sia una c.d.t. pari a metà della tensione di alimentazione, si calcola quindi il valore della medesima R_S con l'espressione:

$R_S = (0{,}5 \times V_a) / I_D$

Quindi, fissata una corrente I_D = 2 mA e una tensione V_a = 26 V, per R_S si ha:

$R_S = (0{,}5 \times 26)/(2 \times 10^{-3}) = 6{,}5\ k\Omega \quad \Rightarrow \quad R_S = 6{,}8\ k\Omega$

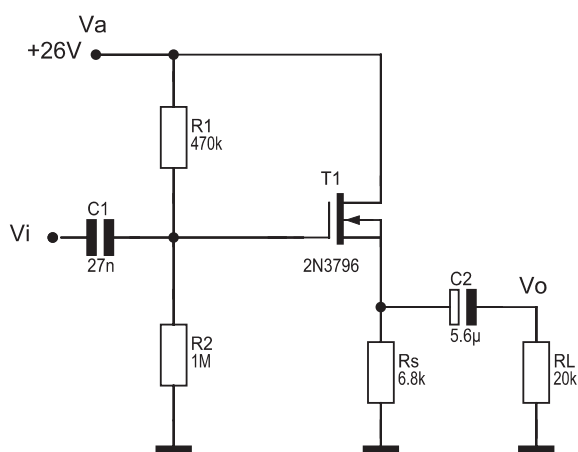



Figura 10
Source follower con Mosfet

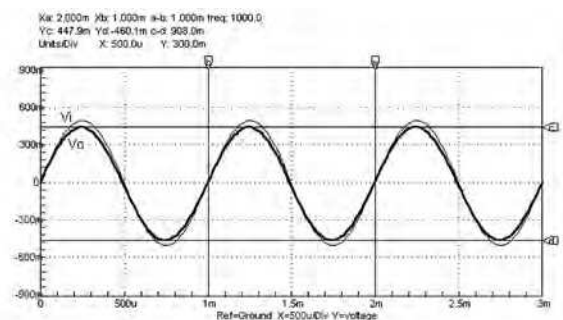



Figura 11
Risposta in ampiezza del source follower con mosfet di cui alla figura 10

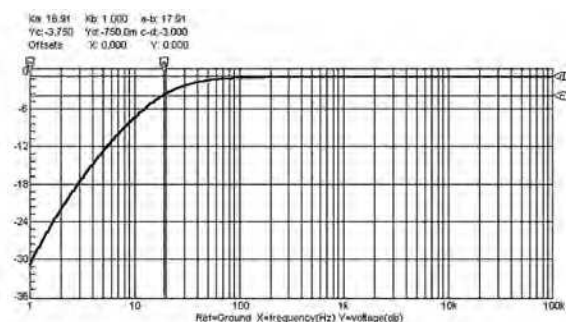



Figura 12
Risposta in frequenza del source follower con mosfet di cui alla figura 10

Si calcola adesso la d. d. p. V_{GS} con l'espressione:

$$V_{GS} = V_T + \sqrt{\frac{I_D}{K}}$$

e quindi la V_{GM} con l'espressione:

$V_{GM} = V_{GS} + R_S\, I_D$

Si ricava allora:

$V_{GM} = 4{,}44 + 13 = 17{,}44\ V$

Attribuito alla R_2 il valore, ad esempio, di 1 MΩ, si calcola la resistenza R_1 con l'espressione:

$R_1 = [(R_2\, V_a) / V_{GM}] - R_2$

Si ottiene:

$R_1=[(106\times 26)/17{,}44]-106=490{,}8\ k\Omega \quad \Rightarrow \quad R_1 = 470\ k\Omega$

Calcolata infine l'impedenza di ingresso coincidente col parallelo delle resistenze R_1 e R_2, con le espressioni già note (v. FE n. 264) si calcolano i valori da attribuire alle capacità C_1 e C_2 per una frequenza di taglio, per esempio, di 20 Hz.

La figura 11 riporta la risposta in ampiezza del source follower. La tensione di uscita ha un valore da picco a picco pari a 908 mV. Essendo il segnale in ingresso pari a 1 V = 1000 mV da picco a picco, per il guadagno *G* si ha:

$G = 908 / 1000 = 0{,}908$

La figura 12 espone quindi la curva della risposta in frequenza. A –3 dB la frequenza di taglio, indicata dalla posizione del marker *a*, è pari a 18,91 Hz.

L'inseguitore di tensione si può infine realizzare con l'amplificatore operazionale. Il voltage follower con op-amp presenta un comportamento ottimale: il segnale di uscita ha ampiezza pressoché pari all'ampiezza del segnale in input e le impedenze di ingresso e di uscita presentano valori facilmente adattabili a qualsiasi impedenza (di sorgente e di carico). Per questa applicazione si rimanda al volume Amplificatori operazionali della collana Conoscere ed usare di Inware Edizioni.

More Info Please!
Inserisci il Codice **267106** alla pagina
www.farelettronica.com/mip
oppure utilizza il **modulo** a pagina **3**

abbonati

SUBITO!!!

e risparmi fino al 37%

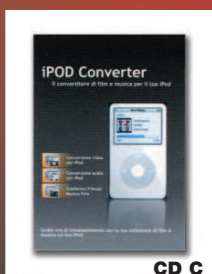
1 — ABBONATI A **Fare Elettronica a soli euro 49,50** PER 11 NUMERI. **Oltre a risparmiare ben 16,50 euro** AVRAI IL PREZZO BLOCCATO PER UN ANNO **e riceverai la rivista comodamente a casa**

2 — ABBONATI A **Fare Elettronica e Firmware a solo euro 89** IL RISPARMIO **sale a ben 43 euro**

3 — CON SOLI **euro 6,50 in più riceverai a scelta** UNO DEI FANTASTICI **CD-ROM** INDICATI QUI A LATO

4 — RINNOVA IL TUO ABBONAMENTO **almeno 3 mesi prima della scadenza** E SCEGLI IL TUO **CD preferito fra quelli a lato, è GRATIS!**

CD A

CD B

CD C

CD A FOTOGRAFIA DIGITALE: un corso completo che guida l'utente nel mondo della fotografia ed in particolare utilizzando le moderne fotocamere digitali.

CD B MASTERIZZARE DVD E FORMATO DivX: tutto sui formati audio e video più diffusi: DVD, VHS, DivX, VCD.

CD B iPOD CONVERTER: il software per convertire i filmati e i file audio in un formato idoneo all'iPOD.

SCOPRI SUL RETRO **di questo coupon** GLI ESCLUSIVI VANTAGGI **che riserviamo agli abbonati**

DIVERSE MODALITA' **per abbonarsi**

Compila, ritaglia e spedisci via fax questo coupon allo **02-66508225**

Spedisci questa pagina in busta chiusa a:
INWARE Edizioni srl Via Cadorna, 27/31 - 20032 Cormano (MI)

Chiamaci allo **02-66504755**

Abbonati on-line sul sito: www.farelettronica.com/abbonamento

Dati PERSONALI

Nome . . . .

Cognome . . . .

Via. . . . . n° . . . .

Cap. . . . . Città . . . . Prov . . . .

Tel . . . . Fax. . . . .

Email . . . .

Ragione Sociale . . . .

P.Iva. . . . . ❍ Fattura

Privacy Ai sensi del Decr. Lgs. 196/2003 la informiamo che i dati trasmessi verranno impiegati coi principali scopi di indagini di mercato e nelle modalità previste dallo stesso, prevalentemente con mezzi informatici. Il conferimento, di norma facoltativo, è obbligatorio per permettere il rapporto commerciale. È in ogni caso fatto diritto dell'interessato esercitare i propri diritti, nei modi previsti dal "Titolo II art. 7" della legge sopra citata, scrivendo a Inware Edizioni srl, Via Cadorna 27 – 20032 Cormano o tramite email a info@inwaredizioni.it

TIPOLOGIA **di abbonamento**

☐ **1** Abbonamento a **11** numeri di **Fare Elettronica**
a soli **EUR 49,50**

☐ **2** Abbonamento a **11** numeri di **Fare Elettronica** + **Firmware**
a soli **EUR 89,00**

☐ **3** Scelgo il CD **A** ❑ **B** ❑ **C** ❑ aggiungendo **EUR 6,50**
al prezzo dell'abbonamento scelto

☐ **4** Scelgo il CD **A** ❑ **B** ❑ **C** ❑ **gratis**
perchè rinnovo il mio abbonamento **3 mesi** prima della scadenza

Modalità DI PAGAMENTO

❍ CARTA DI CREDITO
❑ American Express ❑ Visa ❑ Mastercard
Titolare. . . . .
n°. . . . . scad . . . .

❍ VERSAMENTO
CCP n. 70107552 intestato ad **Inware Edizioni srl**
allegare la ricevuta (o copia) del versamento indicando nella causale:
"Abbonamento Fare Elettronica"

❍ BONIFICO BANCARIO
Appoggiato su Poste Italiane-CIN: I- ABI: 07601- CAB: 01600 - C/C: 000070107552
intestato ad **Inware Edizioni srl**

❍ ALLEGO UN ASSEGNO
intestato ad **Inware Edizioni srl**

Firma . . . .

FE - 267

CONTI CORRENTI POSTALI - Ricevuta di Versamento BancoPosta
€ sul C/C n. 70107552 di Euro
importo in lettere
INTESTATO A: INWARE EDIZIONI SRL
ESEGUITO DA: VIA - PIAZZA CAP LOCALITÀ
BOLLO DELL'UFF. POSTALE
CAUSALE
AVVERTENZE
Il Bollettino deve essere compilato in ogni sua parte (con inchiostro nero o blu) e non deve recare abrasioni, correzioni o cancellature.
La causale è obbligatoria per i versamenti a favore delle Pubbliche Amministrazioni.
Le informazioni richieste vanno riportate in modo identico in ciascuna delle parti di cui si compone il bollettino.

CONTI CORRENTI POSTALI - Ricevuta di Accredito BancoPosta
€ sul C/C n. 70107552 di Euro
TD 451 importo in lettere
INTESTATO A: INWARE EDIZIONI SRL
CAUSALE
ESEGUITO DA: VIA - PIAZZA CAP LOCALITÀ
BOLLO DELL'UFF. POSTALE codice bancoposta
IMPORTANTE: NON SCRIVERE NELLA ZONA SOTTOSTANTE
importo in euro numero conto td
70107552 < 451>
AUT. DB/SISB/E 20017 DEL 11.04.2006

Il vero valore lo riconosci a prima vista

Codice MIP **267116**